Probabilistic Inference (CO-493)

Imperial College London

# Variational Inference

**Marc Deisenroth**

Department of Computing
Imperial College London

m.deisenroth@imperial.ac.uk

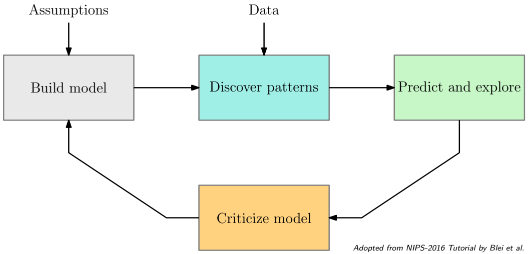
February 19, 2019

# Learning Material

- Pattern Recognition and Machine Learning, Chapter 10 (Bishop, 2006)
- Machine Learning: A Probabilistic Perspective, Chapter 21 (Murphy, 2012)
- Variational Inference: A Review for Statisticians (Blei et al., 2017)
- NIPS-2016 Tutorial by Blei, Ranganath, Mohamed
  `https://nips.cc/Conferences/2016/Schedule?showEvent=6199`
- Tutorials by S. Mohamed
  `http://shakirm.com/papers/VITutorial.pdf`
  `http://shakirm.com/slides/MLSS2018-Madrid-ProbThinking.pdf`

# Overview

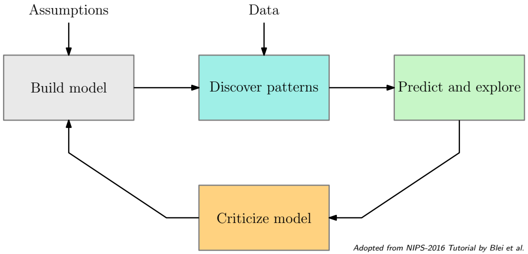
# Probabilistic Pipeline



Adopted from NIPS-2016 Tutorial by Blei et al.

- Use knowledge and assumptions about the data to build a model
- Use model and data to discover patterns
- Predict and explore
- Criticize/revise the model

# Probabilistic Pipeline




- Use knowledge and assumptions about the data to build a model
- Use model and data to discover patterns
- Predict and explore
- Criticize/revise the model

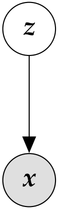
▶▶ **Inference is the key algorithmic problem**: What does the model say about the data?

▶▶ **Goal: general and scalable approaches to inference**

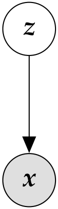
# Probabilistic Machine Learning

- **Probabilistic model:** Joint distribution of latent variables $\boldsymbol{z}$ and observed variables $\boldsymbol{x}$ (data):

$$p(\boldsymbol{x}, \boldsymbol{z})$$



# Probabilistic Machine Learning



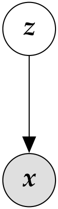

- **Probabilistic model:** Joint distribution of latent variables $\boldsymbol{z}$ and observed variables $\boldsymbol{x}$ (data):

$$p(\boldsymbol{x}, \boldsymbol{z})$$

- **Inference**: Learning about the unknowns $\boldsymbol{z}$ through the posterior distribution

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, \boldsymbol{z})}{p(\boldsymbol{x})}, \qquad p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z}$$

# Probabilistic Machine Learning




- **Probabilistic model:** Joint distribution of latent variables $\boldsymbol{z}$ and observed variables $\boldsymbol{x}$ (data):

$$p(\boldsymbol{x}, \boldsymbol{z})$$

- **Inference**: Learning about the unknowns $\boldsymbol{z}$ through the posterior distribution

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, \boldsymbol{z})}{p(\boldsymbol{x})}, \qquad p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}$$

- Normally: Denominator (marginal likelihood/evidence) intractable (i.e., we cannot compute the integral)
  ▸▸ Approximate inference to get the posterior

# Some Options for Posterior Inference

- Markov Chain Monte Carlo (to sample from the posterior)
- Laplace approximation
- Expectation propagation (Minka, 2001)
- **Variational inference** (Jordan et al., 1999)

# Variational Inference

- Variational inference is the most scalable inference method available (at the moment)
- Can handle (arbitrarily) large datasets

# Variational Inference

- Variational inference is the most scalable inference method available (at the moment)
- Can handle (arbitrarily) large datasets
- Applications include:
  - Topic modeling (Hoffman et al., 2013)
  - Community detection (Gopalan & Blei, 2013)
  - Genetic analysis (Gopalan et al., 2016)
  - Reinforcement learning (e.g., Eslami et al., 2016)
  - Neuroscience analysis (Manning et al., 2014)
  - Compression and content generation (Gregor et al., 2016)
  - Traffic analysis (Kucukelbir et al., 2016; Salimbeni & Deisenroth, 2017)

# Overview

# Key Idea: Approximation by Optimization

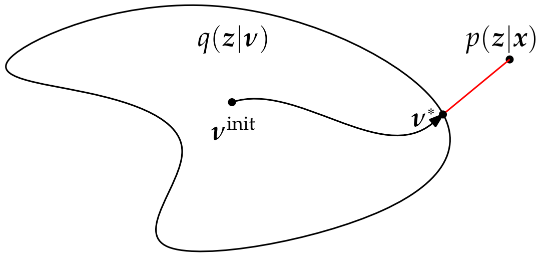



*Figure adopted from Blei et al.'s NIPS-2016 tutorial*

# Key Idea: Approximation by Optimization

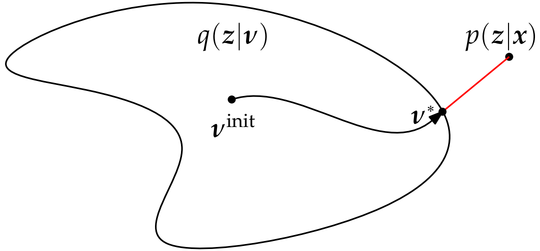



*Figure adopted from Blei et al.'s NIPS-2016 tutorial*

- Find approximation of a probability distribution (e.g., posterior) by optimization:
  1. Define an objective function
  2. Define a (parametrized) family of approximating distributions $q_{\boldsymbol{\nu}}$
  3. Optimize objective function w.r.t. variational parameters $\boldsymbol{\nu}$
- Inference ⏩ Optimization

# Overview

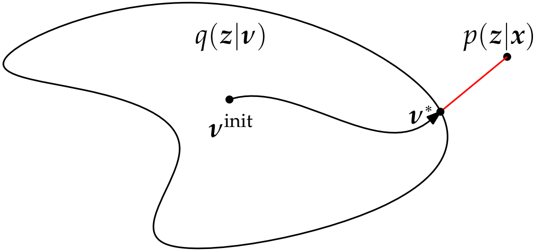



*Figure adopted from Blei et al.'s NIPS-2016 tutorial*

- Find approximation of a probability distribution (e.g., posterior) by optimization:
  1. **Define an objective function**
  2. Define a (parametrized) family of approximating distributions $q_{\boldsymbol{\nu}}$
  3. Optimize objective function w.r.t. variational parameters $\boldsymbol{\nu}$
- Inference ⏩ Optimization

# Overview

# Some Useful Quantities

- Kullback-Leibler divergence

$$\mathrm{KL}(q(\boldsymbol{x})\|p(\boldsymbol{x})) = \int q(\boldsymbol{x}) \log \frac{q(\boldsymbol{x})}{p(\boldsymbol{x})} d\boldsymbol{x}$$

$$= \mathbb{E}_q \left[ \log \frac{q(\boldsymbol{x})}{p(\boldsymbol{x})} \right] = \mathbb{E}_q[\log q(\boldsymbol{x})] - \mathbb{E}_q[\log p(\boldsymbol{x})]$$

# Some Useful Quantities

- Kullback-Leibler divergence

$$\mathrm{KL}(q(\boldsymbol{x})\|p(\boldsymbol{x})) = \int q(\boldsymbol{x}) \log \frac{q(\boldsymbol{x})}{p(\boldsymbol{x})} d\boldsymbol{x}$$
$$= \mathbb{E}_q\left[\log \frac{q(\boldsymbol{x})}{p(\boldsymbol{x})}\right] = \mathbb{E}_q[\log q(\boldsymbol{x})] - \mathbb{E}_q[\log p(\boldsymbol{x})]$$

- Differential entropy

$$\mathrm{H}[q(\boldsymbol{x})] = -\mathbb{E}_q[\log q(\boldsymbol{x})] = -\int q(\boldsymbol{x}) \log q(\boldsymbol{x}) d\boldsymbol{x}$$

# Optimization Objective

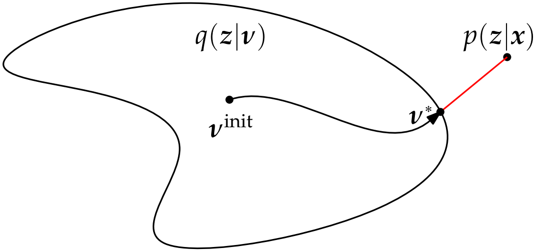



*Figure adopted from Blei et al.'s NIPS-2016 tutorial*

- Need to compare distributions (variational approximation $q$, true posterior $p$)

  ▶▶ Kullback-Leibler divergence

# Optimization Objective

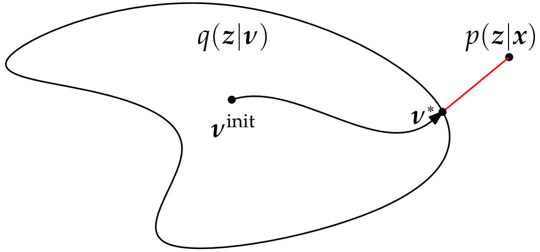



*Figure adopted from Blei et al.'s NIPS-2016 tutorial*

- Need to compare distributions (variational approximation $q$, true posterior $p$)

  ⏩ Kullback-Leibler divergence
- Find variational parameters $\boldsymbol{\nu}$ by minimizing the KL divergence $\text{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}|\boldsymbol{x}))$ between the variational approximation $q$ and the true posterior.

# Optimization Objective (2)

- Minimize the KL divergence $\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}|\boldsymbol{x}))$ between the variational approximation $q$ and the true posterior.
  ▶▶ $q(\boldsymbol{z}|\boldsymbol{\nu}) = p(\boldsymbol{z}|\boldsymbol{x})$ is the optimal solution

# Optimization Objective (2)

- Minimize the KL divergence $\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}|\boldsymbol{x}))$ between the variational approximation $q$ and the true posterior.
  ⏩ $q(\boldsymbol{z}|\boldsymbol{\nu}) = p(\boldsymbol{z}|\boldsymbol{x})$ is the optimal solution

$$\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x})) = \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z}|\boldsymbol{x})]$$

# Optimization Objective (2)

- Minimize the KL divergence $\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}|\boldsymbol{x}))$ between the variational approximation $q$ and the true posterior.
  ▶▶ $q(\boldsymbol{z}|\boldsymbol{\nu}) = p(\boldsymbol{z}|\boldsymbol{x})$ is the optimal solution

$$\begin{aligned}\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x})) &= \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z}|\boldsymbol{x})] \\ &= \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \log p(\boldsymbol{x})\end{aligned}$$

# Optimization Objective (2)

- Minimize the KL divergence $\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}|\boldsymbol{x}))$ between the variational approximation $q$ and the true posterior.
  ▸▸ $q(\boldsymbol{z}|\boldsymbol{\nu}) = p(\boldsymbol{z}|\boldsymbol{x})$ is the optimal solution

$$\begin{aligned}
\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x})) &= \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z}|\boldsymbol{x})] \\
&= \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \log p(\boldsymbol{x}) \\
&= \mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z})) - \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \mathrm{const}
\end{aligned}$$

# Optimization Objective (2)

- Minimize the KL divergence $\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}|\boldsymbol{x}))$ between the variational approximation $q$ and the true posterior.
  ▶▶ $q(\boldsymbol{z}|\boldsymbol{\nu}) = p(\boldsymbol{z}|\boldsymbol{x})$ is the optimal solution

$$\begin{aligned}
\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x})) &= \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z}|\boldsymbol{x})] \\
&= \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \log p(\boldsymbol{x}) \\
&= \mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z})) - \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \mathrm{const}
\end{aligned}$$

- Not required to know the unknown posterior $p(\boldsymbol{z}|\boldsymbol{x})$

# Optimization Objective (2)

- Minimize the KL divergence $\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}|\boldsymbol{x}))$ between the variational approximation $q$ and the true posterior.
  ▸▸ $q(\boldsymbol{z}|\boldsymbol{\nu}) = p(\boldsymbol{z}|\boldsymbol{x})$ is the optimal solution

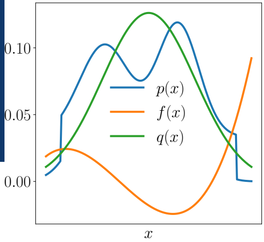
$$\begin{aligned}\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x})) &= \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z}|\boldsymbol{x})] \\ &= \mathbb{E}_q[\log q(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \log p(\boldsymbol{x}) \\ &= \mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z})) - \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \mathrm{const}\end{aligned}$$

- Not required to know the unknown posterior $p(\boldsymbol{z}|\boldsymbol{x})$
- Minimizing $\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x}))$ is equivalent to maximizing a lower bound on the marginal likelihood (ELBO)

# Evidence Lower Bound

# Importance Sampling

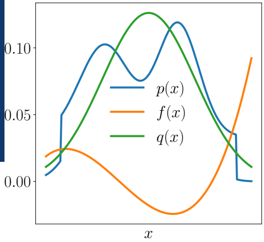
**Key idea:** Transform an intractable integral into an expectation under a simpler distribution $q$ (proposal distribution):




$$\mathbb{E}_p[f(\boldsymbol{x})] = \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$$

# Importance Sampling

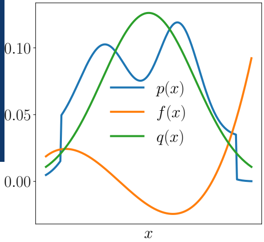
**Key idea:** Transform an intractable integral into an expectation under a simpler distribution $q$ (proposal distribution):




$$\mathbb{E}_p[f(\boldsymbol{x})] = \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$$
$$= \int f(\boldsymbol{x})p(\boldsymbol{x})\frac{q(\boldsymbol{x})}{q(\boldsymbol{x})}d\boldsymbol{x}$$

# Importance Sampling

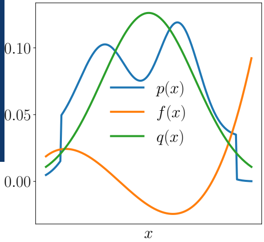
**Key idea:** Transform an intractable integral into an expectation under a simpler distribution $q$ (proposal distribution):




$$\mathbb{E}_p[f(\boldsymbol{x})] = \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$$
$$= \int f(\boldsymbol{x})p(\boldsymbol{x})\frac{q(\boldsymbol{x})}{q(\boldsymbol{x})}d\boldsymbol{x} = \int f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}q(\boldsymbol{x})d\boldsymbol{x}$$

# Importance Sampling

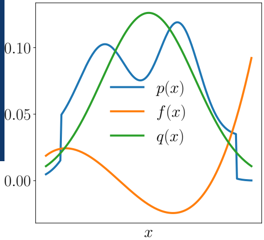
**Key idea:** Transform an intractable integral into an expectation under a simpler distribution $q$ (proposal distribution):




$$
\begin{aligned}
\mathbb{E}_p[f(\boldsymbol{x})] &= \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x} \\
&= \int f(\boldsymbol{x})p(\boldsymbol{x})\frac{q(\boldsymbol{x})}{q(\boldsymbol{x})}d\boldsymbol{x} = \int f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}q(\boldsymbol{x})d\boldsymbol{x} \\
&= \mathbb{E}_q\left[f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right]
\end{aligned}
$$

# Importance Sampling

**Key idea:** Transform an intractable integral into an expectation under a simpler distribution $q$ (proposal distribution):



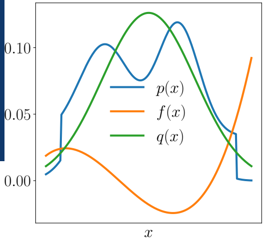

$$\begin{aligned}
\mathbb{E}_p[f(\boldsymbol{x})] &= \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x} \\
&= \int f(\boldsymbol{x})p(\boldsymbol{x})\frac{q(\boldsymbol{x})}{q(\boldsymbol{x})}d\boldsymbol{x} = \int f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}q(\boldsymbol{x})d\boldsymbol{x} \\
&= \mathbb{E}_q\left[f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right]
\end{aligned}$$

If we choose $q$ in a way that we can easily sample from it, we can approximate this last expectation by Monte Carlo:

$$\mathbb{E}_q\left[f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right] \approx \frac{1}{S}\sum_{s=1}^{S} f(\boldsymbol{x}^{(s)})\frac{p(\boldsymbol{x}^{(s)})}{q(\boldsymbol{x}^{(s)})} \quad , \quad \boldsymbol{x}^{(s)} \sim q(\boldsymbol{x})$$

# Importance Sampling

**Key idea:** Transform an intractable integral into an expectation under a simpler distribution $q$ (proposal distribution):




$$\mathbb{E}_p[f(\boldsymbol{x})] = \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$$
$$= \int f(\boldsymbol{x})p(\boldsymbol{x})\frac{q(\boldsymbol{x})}{q(\boldsymbol{x})}d\boldsymbol{x} = \int f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}q(\boldsymbol{x})d\boldsymbol{x}$$
$$= \mathbb{E}_q\left[f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right]$$

If we choose $q$ in a way that we can easily sample from it, we can approximate this last expectation by Monte Carlo:

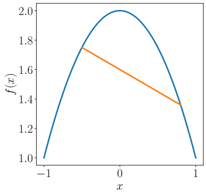
$$\mathbb{E}_q\left[f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right] \approx \frac{1}{S}\sum_{s=1}^{S} f(\boldsymbol{x}^{(s)})\frac{p(\boldsymbol{x}^{(s)})}{q(\boldsymbol{x}^{(s)})} = \frac{1}{S}\sum_{s=1}^{S} w_s f(\boldsymbol{x}^{(s)}), \quad \boldsymbol{x}^{(s)} \sim q(\boldsymbol{x})$$

# Importance Sampling: Properties

- Unbiased estimate of the expectation
- Many draws from posterior needed, especially in high dimensions
- Good for evaluating integrals, but we don't know much about the posterior distribution
- Degeneracy

# Jensen's Inequality

An important result from convex analysis:




**Jensen's Inequality**

For concave functions $f$:

$$f(\mathbb{E}[z]) \geqslant \mathbb{E}[f(z)]$$

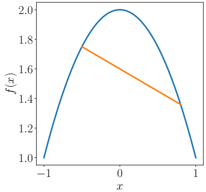
# Jensen's Inequality

An important result from convex analysis:

**Jensen's Inequality**

For concave functions $f$:

$$f(\mathbb{E}[z]) \geqslant \mathbb{E}[f(z)]$$




Logarithms are concave. Therefore:

$$\log \mathbb{E}[f(z)] = \log \int f(z)p(z)dz \geqslant \int p(z) \log f(z)dz = \mathbb{E}[\log f(z)]$$

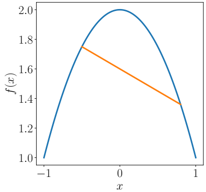
# Jensen's Inequality

An important result from convex analysis:

**Jensen's Inequality**

For concave functions $f$:

$$f(\mathbb{E}[z]) \geqslant \mathbb{E}[f(z)]$$




Logarithms are concave. Therefore:

$$\log \mathbb{E}[f(z)] = \log \int f(z)p(z)dz \geqslant \int p(z) \log f(z)dz = \mathbb{E}[\log f(z)]$$

Idea: For computing the marginal likelihood, use Jensen's inequality instead of MCMC

# From Importance Sampling to Variational Inference

Look at log-marginal likelihood (log-evidence):

$$\log p(\boldsymbol{x}) = \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z}$$

# From Importance Sampling to Variational Inference

Look at log-marginal likelihood (log-evidence):

$$\begin{aligned}\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \\ &= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\frac{q(\boldsymbol{z})}{q(\boldsymbol{z})}d\boldsymbol{z}\end{aligned}$$

# From Importance Sampling to Variational Inference

Look at log-marginal likelihood (log-evidence):

$$\begin{aligned}\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}|z)p(z)dz \\ &= \log \int p(\boldsymbol{x}|z)p(z)\frac{q(z)}{q(z)}dz \\ &= \log \int p(\boldsymbol{x}|z)\frac{p(z)}{q(z)}q(z)dz\end{aligned}$$

# From Importance Sampling to Variational Inference

Look at log-marginal likelihood (log-evidence):

$$\begin{aligned}\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \\ &= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\frac{q(\boldsymbol{z})}{q(\boldsymbol{z})}d\boldsymbol{z} \\ &= \log \int p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}q(\boldsymbol{z})d\boldsymbol{z} \\ &= \log \mathbb{E}_q\left[p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right]\end{aligned}$$

# From Importance Sampling to Variational Inference

Look at log-marginal likelihood (log-evidence):

$$\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \\
&= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\frac{q(\boldsymbol{z})}{q(\boldsymbol{z})}d\boldsymbol{z} \\
&= \log \int p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}q(\boldsymbol{z})d\boldsymbol{z} \\
&= \log \mathbb{E}_q\left[p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right] \\
&\geqslant \mathbb{E}_q \log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right)
\end{aligned}$$

# From Importance Sampling to Variational Inference

Look at log-marginal likelihood (log-evidence):

$$\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \\
&= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\frac{q(\boldsymbol{z})}{q(\boldsymbol{z})}d\boldsymbol{z} \\
&= \log \int p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}q(\boldsymbol{z})d\boldsymbol{z} \\
&= \log \mathbb{E}_q\left[p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right] \\
&\geqslant \mathbb{E}_q \log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right) \\
&= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \mathbb{E}_q\left[\log\left(\frac{q(\boldsymbol{z})}{p(\boldsymbol{z})}\right)\right]
\end{aligned}$$

# From Importance Sampling to Variational Inference

Look at log-marginal likelihood (log-evidence):

$$
\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \\
&= \log \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\frac{q(\boldsymbol{z})}{q(\boldsymbol{z})}d\boldsymbol{z} \\
&= \log \int p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}q(\boldsymbol{z})d\boldsymbol{z} \\
&= \log \mathbb{E}_q\left[p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right] \\
&\geqslant \mathbb{E}_q \log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right) \\
&= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \mathbb{E}_q\left[\log\left(\frac{q(\boldsymbol{z})}{p(\boldsymbol{z})}\right)\right] \\
&= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}))
\end{aligned}
$$

# Evidence Lower Bound (ELBO)

- We just lower-bounded the evidence (marginal likelihood):

$$\log p(\boldsymbol{x}) \geqslant \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z})) =: \mathrm{ELBO}$$

# Evidence Lower Bound (ELBO)

- We just lower-bounded the evidence (marginal likelihood):

$$\log p(\boldsymbol{x}) \geqslant \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z})) =: \mathrm{ELBO}$$

- Data-fit term (expected log-likelihood): Measures how well samples from $q(\boldsymbol{z})$ explain the data ("reconstruction cost").
  ⏩ Place $q$'s mass on the MAP estimate.
- Regularizer: Variational posterior $q(\boldsymbol{z})$ should not differ much from the prior $p(\boldsymbol{z})$

# ELBO and KL Divergence

- Maximizing the ELBO w.r.t. the variational parameters $\boldsymbol{\nu}$ is equivalent to minimizing $\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x}))$, where $p(\boldsymbol{z}|\boldsymbol{x})$ is the true posterior

# ELBO and KL Divergence

- Maximizing the ELBO w.r.t. the variational parameters $\boldsymbol{\nu}$ is equivalent to minimizing $\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x}))$, where $p(\boldsymbol{z}|\boldsymbol{x})$ is the true posterior

$$\begin{aligned}\mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x})) &= \mathrm{KL}(q(\boldsymbol{z})\|p(\boldsymbol{z})) - \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \text{const} \\ &= -\text{ELBO} + \text{const}\end{aligned}$$

# Optimizing the ELBO

$$\mathcal{F}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})]}_{\text{data-fit}} - \underbrace{\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}))}_{\text{regularizer}}$$
$$= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z})] + \mathrm{H}[q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Choice of the variational distribution $q(\boldsymbol{z}|\boldsymbol{\nu})$ ⏩ Parametrization

# Optimizing the ELBO

$$\mathcal{F}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})]}_{\text{data-fit}} - \underbrace{\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}))}_{\text{regularizer}}$$
$$= \mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z})] + \mathrm{H}[q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Choice of the variational distribution $q(\boldsymbol{z}|\boldsymbol{\nu})$ ⏩ Parametrization
- Computational challenges:

# Optimizing the ELBO

$$\mathcal{F}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})]}_{\text{data-fit}} - \underbrace{\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}))}_{\text{regularizer}}$$
$$= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z})] + \mathrm{H}[q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Choice of the variational distribution $q(\boldsymbol{z}|\boldsymbol{\nu})$ ▸▸ Parametrization
- Computational challenges:
  - Compute expectation

# Optimizing the ELBO

$$\mathcal{F}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})]}_{\text{data-fit}} - \underbrace{\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}))}_{\text{regularizer}}$$
$$= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z})] + \mathrm{H}[q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Choice of the variational distribution $q(\boldsymbol{z}|\boldsymbol{\nu})$ ⏩ Parametrization
- Computational challenges:
  - Compute expectation
  - Compute gradients $d\mathcal{F}/d\boldsymbol{\nu}$ for variational learning

# Optimizing the ELBO

$$\mathcal{F}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})]}_{\text{data-fit}} - \underbrace{\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}))}_{\text{regularizer}}$$
$$= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z})] + \mathrm{H}[q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Choice of the variational distribution $q(\boldsymbol{z}|\boldsymbol{\nu})$ ▶▶ Parametrization
- Computational challenges:
  - Compute expectation
  - Compute gradients $d\mathcal{F}/d\boldsymbol{\nu}$ for variational learning
  - ELBO is non-convex ▶▶ Local optima

# Overview

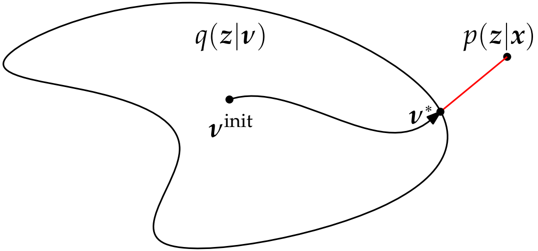



*Figure adopted from Blei et al.'s NIPS-2016 tutorial*

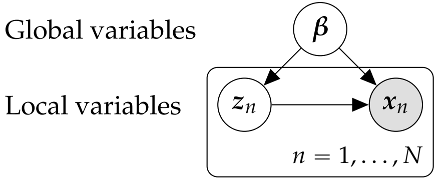
- Find approximation of a probability distribution (e.g., posterior) by optimization:
  1. Define an objective function
  2. **Define a (parametrized) family of approximating distributions** $q_{\boldsymbol{\nu}}$
  3. Optimize objective function w.r.t. variational parameters $\boldsymbol{\nu}$
- Inference ⏩ Optimization

# Roadmap I

1. Define a generic class of conditionally conjugate models
2. Classical mean-field variational inference
3. Stochastic variational inference ⏩ Scales to massive data

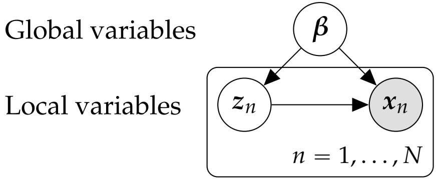
# Overview

# Model Class



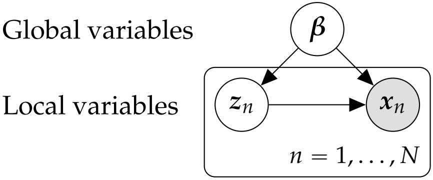

- All unknown parameters are described by random variables
  - Global random variables $\boldsymbol{\beta}$, which control all the data
  - Local random variables $\boldsymbol{z}_n$, which are local to individual data points $\boldsymbol{x}_n$

# Model Class



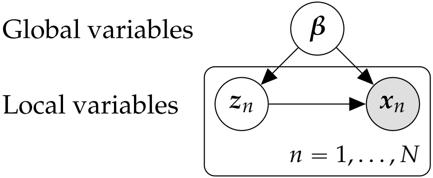

- All unknown parameters are described by random variables
  - Global random variables $\beta$, which control all the data
  - Local random variables $z_n$, which are local to individual data points $x_n$
- Example: Gaussian mixture model

# Model Class



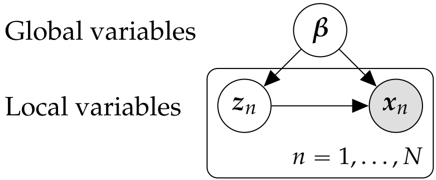

- All unknown parameters are described by random variables
  - Global random variables $\boldsymbol{\beta}$, which control all the data
  - Local random variables $\boldsymbol{z}_n$, which are local to individual data points $\boldsymbol{x}_n$
- Example: Gaussian mixture model
  - Global: means, covariances, weights $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k$
  - Local: binary assignments $\boldsymbol{z}_n$

# Probabilistic Model



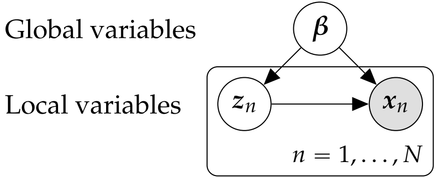

- Observations/data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$
- $n$th data point depends only on local $\boldsymbol{z}_n$ and global $\boldsymbol{\beta}$
- Joint distribution:

# Probabilistic Model




- Observations/data $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$
- $n$th data point depends only on local $\boldsymbol{z}_n$ and global $\boldsymbol{\beta}$
- Joint distribution: $p(\boldsymbol{\beta}, \boldsymbol{z}_{1:N}, \boldsymbol{x}_{1:N}) = p(\boldsymbol{\beta}) \prod_{n=1}^{N} p(\boldsymbol{z}_n, \boldsymbol{x}_n | \boldsymbol{\beta})$

# Probabilistic Model

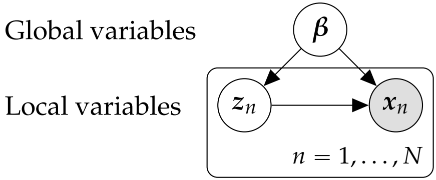



- Observations/data $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$
- $n$th data point depends only on local $\boldsymbol{z}_n$ and global $\boldsymbol{\beta}$
- Joint distribution: $p(\boldsymbol{\beta}, \boldsymbol{z}_{1:N}, \boldsymbol{x}_{1:N}) = p(\boldsymbol{\beta}) \prod_{n=1}^{N} p(\boldsymbol{z}_n, \boldsymbol{x}_n | \boldsymbol{\beta})$

**Objective**

Compute posterior distribution of all unknowns: $p(\boldsymbol{\beta}, \boldsymbol{z}_{1:N} | \boldsymbol{x}_{1:N})$.

# Complete Conditional

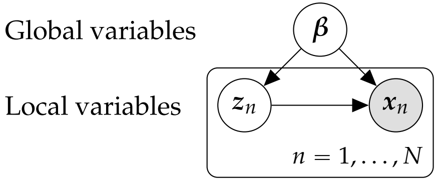



- **Complete conditional:** Conditional of a single latent variable given the observations and all other latent variables

$$p(\boldsymbol{z}_n | \boldsymbol{\beta}, \boldsymbol{x}_n)$$

$$p(\boldsymbol{\beta} | \boldsymbol{z}_{1:N}, \boldsymbol{x}_{1:N})$$

# Complete Conditional




- **Complete conditional:** Conditional of a single latent variable given the observations and all other latent variables

$$p(\boldsymbol{z}_n | \boldsymbol{\beta}, \boldsymbol{x}_n)$$
$$p(\boldsymbol{\beta} | \boldsymbol{z}_{1:N}, \boldsymbol{x}_{1:N})$$

- Assume that each complete conditional is a member of the exponential family (Bernoulli, Beta, Gamma, Gaussian, ...)
  ▸▸ Conditionally conjugate models

# Generic Class of Models: Examples

- Bayesian mixture models
- Hidden Markov models
- Factor analysis
- Principal component analysis
- Linear regression

# Approximating Distributions


$q(z|x) = p(z|x)$

$q(z|x) = \prod_i q_i(z_i)$

# Approximating Distributions


- Specifying the class of posteriors is closely related to specifying a model of the data
  ▶▶ We have a lot of flexibility

# Approximating Distributions

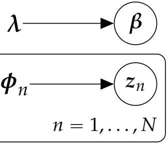

- Specifying the class of posteriors is closely related to specifying a model of the data
  ⏩ We have a lot of flexibility
- Generally:
  - Build expressive class of posteriors (no overfitting problems)
  - Maintain computational efficiency ⏩ Scalability

# Overview

# Mean-Field Approximation


- Assume conditionally conjugate model
- Fully factorized (mean field) approximation:

$$q(\boldsymbol{\beta}, \boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{\beta}|\boldsymbol{\lambda}) \prod_{n=1}^{N} q(\boldsymbol{z}_n|\boldsymbol{\phi}_n)$$



# Fully Factorized Distribution

- Fully factorized (mean field) approximation:

$$q(\boldsymbol{\beta}, \boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{\beta}|\boldsymbol{\lambda}) \prod_{n=1}^{N} q(\boldsymbol{z}_n|\boldsymbol{\phi}_n), \quad \boldsymbol{\nu} = \{\boldsymbol{\lambda}, \boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_N\}$$

# Fully Factorized Distribution

- Fully factorized (mean field) approximation:

$$q(\boldsymbol{\beta}, \boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{\beta}|\boldsymbol{\lambda}) \prod_{n=1}^{N} q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) \,, \quad \boldsymbol{\nu} = \{\boldsymbol{\lambda}, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N\}$$

- All latent variables are independent and governed by their own variational parameters

# Fully Factorized Distribution

- Fully factorized (mean field) approximation:

$$q(\boldsymbol{\beta}, \boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{\beta}|\boldsymbol{\lambda}) \prod_{n=1}^{N} q(\boldsymbol{z}_n|\boldsymbol{\phi}_n), \quad \boldsymbol{\nu} = \{\boldsymbol{\lambda}, \boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_N\}$$

- All latent variables are independent and governed by their own variational parameters
- Assume each factor is in the same exponential family as the model's complete conditional

# Fully Factorized Distribution

- Fully factorized (mean field) approximation:

$$q(\boldsymbol{\beta}, \boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{\beta}|\boldsymbol{\lambda}) \prod_{n=1}^{N} q(\boldsymbol{z}_n|\boldsymbol{\phi}_n), \quad \boldsymbol{\nu} = \{\boldsymbol{\lambda}, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N\}$$

- All latent variables are independent and governed by their own variational parameters
- Assume each factor is in the same exponential family as the model's complete conditional
- The $q$-factors do not depend on the data

# Fully Factorized Distribution

- Fully factorized (mean field) approximation:

$$q(\boldsymbol{\beta}, \boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{\beta}|\boldsymbol{\lambda}) \prod_{n=1}^{N} q(\boldsymbol{z}_n|\boldsymbol{\phi}_n), \quad \boldsymbol{\nu} = \{\boldsymbol{\lambda}, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N\}$$

- All latent variables are independent and governed by their own variational parameters
- Assume each factor is in the same exponential family as the model's complete conditional
- The $q$-factors do not depend on the data
- ELBO connects this family to the data

# Fully Factorized Distribution

- Fully factorized (mean field) approximation:

$$q(\boldsymbol{\beta}, \boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{\beta}|\boldsymbol{\lambda}) \prod_{n=1}^{N} q(\boldsymbol{z}_n|\boldsymbol{\phi}_n), \quad \boldsymbol{\nu} = \{\boldsymbol{\lambda}, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N\}$$

- All latent variables are independent and governed by their own variational parameters
- Assume each factor is in the same exponential family as the model's complete conditional
- The $q$-factors do not depend on the data
- ELBO connects this family to the data
- Maximize the ELBO w.r.t. variational parameters $\boldsymbol{\nu}$

# Optimizing in Turn

- Independent latent variables: $q(\boldsymbol{z}) = \prod_n q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) = \prod_n q_n(\boldsymbol{z}_n)$
  - ▸▸ Optimize variational parameters in turn

# Optimizing in Turn

- Independent latent variables: $q(\boldsymbol{z}) = \prod_n q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) = \prod_n q_n(\boldsymbol{z}_n)$
  - ▶▶ Optimize variational parameters in turn

$$\text{ELBO} = \mathbb{E}_q\left[\log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right)\right]$$

# Optimizing in Turn

- Independent latent variables: $q(\boldsymbol{z}) = \prod_n q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) = \prod_n q_n(\boldsymbol{z}_n)$
  - ⏩ Optimize variational parameters in turn

$$\text{ELBO} = \mathbb{E}_q\left[\log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right)\right] = \mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q(\boldsymbol{z})]$$

# Optimizing in Turn

- Independent latent variables: $q(\boldsymbol{z}) = \prod_n q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) = \prod_n q_n(\boldsymbol{z}_n)$

  ⏩ Optimize variational parameters in turn

$$\text{ELBO} = \mathbb{E}_q\left[\log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right)\right] = \mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q(\boldsymbol{z})]$$

$$= \mathbb{E}_{q_n}\Big[\underbrace{\mathbb{E}_{i\neq n}[\log p(\boldsymbol{x},\boldsymbol{z})]}_{=:\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)}\Big] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

# Optimizing in Turn

- Independent latent variables: $q(\boldsymbol{z}) = \prod_n q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) = \prod_n q_n(\boldsymbol{z}_n)$
  - ⏩ Optimize variational parameters in turn

$$\text{ELBO} = \mathbb{E}_q\left[\log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right)\right] = \mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q(\boldsymbol{z})]$$

$$= \mathbb{E}_{q_n}\Big[\underbrace{\mathbb{E}_{i\neq n}[\log p(\boldsymbol{x},\boldsymbol{z})]}_{=:\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)}\Big] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

$$= \mathbb{E}_{q_n}[\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

# Optimizing in Turn

- Independent latent variables: $q(\boldsymbol{z}) = \prod_n q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) = \prod_n q_n(\boldsymbol{z}_n)$
  - ▸▸ Optimize variational parameters in turn

$$\text{ELBO} = \mathbb{E}_q\left[\log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right)\right] = \mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q(\boldsymbol{z})]$$

$$= \mathbb{E}_{q_n}\Big[\underbrace{\mathbb{E}_{i\neq n}[\log p(\boldsymbol{x},\boldsymbol{z})]}_{=:\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)}\Big] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

$$= \mathbb{E}_{q_n}[\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

- Fix $q_{i\neq n}$ and optimize factor $q_n$:

# Optimizing in Turn

- Independent latent variables: $q(\boldsymbol{z}) = \prod_n q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) = \prod_n q_n(\boldsymbol{z}_n)$

  ⏩ Optimize variational parameters in turn

$$\text{ELBO} = \mathbb{E}_q\left[\log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right)\right] = \mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q(\boldsymbol{z})]$$

$$= \mathbb{E}_{q_n}\Big[\underbrace{\mathbb{E}_{i\neq n}[\log p(\boldsymbol{x},\boldsymbol{z})]}_{=:\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)}\Big] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

$$= \mathbb{E}_{q_n}[\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

- Fix $q_{i\neq n}$ and optimize factor $q_n$:

$$\text{ELBO}(q_n) = \mathbb{E}_{q_n}[\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] + \text{const}$$

# Optimizing in Turn

- Independent latent variables: $q(\boldsymbol{z}) = \prod_n q(\boldsymbol{z}_n|\boldsymbol{\phi}_n) = \prod_n q_n(\boldsymbol{z}_n)$

  ▶▶ Optimize variational parameters in turn

$$\text{ELBO} = \mathbb{E}_q\left[\log\left(p(\boldsymbol{x}|\boldsymbol{z})\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}\right)\right] = \mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z}) - \log q(\boldsymbol{z})]$$

$$= \mathbb{E}_{q_n}\Big[\underbrace{\mathbb{E}_{i\neq n}[\log p(\boldsymbol{x},\boldsymbol{z})]}_{=:\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)}\Big] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

$$= \mathbb{E}_{q_n}[\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] - \sum_{i\neq n}\mathbb{E}_{q_i}[\log q_i(\boldsymbol{z}_i)]$$

- Fix $q_{i\neq n}$ and optimize factor $q_n$:

$$\text{ELBO}(q_n) = \mathbb{E}_{q_n}[\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)] - \mathbb{E}_{q_n}[\log q_n(\boldsymbol{z}_n)] + \text{const}$$

$$= \mathbb{E}_{q_n}\left[\log\frac{\exp(\hat{p}(\boldsymbol{x},\boldsymbol{z}_n))}{q_n(\boldsymbol{z}_n)}\right] = -\text{KL}(q_n(\boldsymbol{z}_n)\|\exp(\hat{p}(\boldsymbol{x},\boldsymbol{z}_n)))$$

# Optimal Factors

$$\text{ELBO}(q_n) = -\text{KL}(q_n(\boldsymbol{z}_n) \| \exp(\hat{p}(\boldsymbol{x}, \boldsymbol{z}_n)))$$

- Maximizing the ELBO w.r.t. $q_n$ is equivalent to minimizing $\text{KL}(q_n(\boldsymbol{z}_n) \| \exp(\hat{p}(\boldsymbol{x}, \boldsymbol{z}_n)))$

▶▶ $\log q_n^*(\boldsymbol{z}_n) = \hat{p}(\boldsymbol{x}, \boldsymbol{z}_n) = \mathbb{E}_{q_{i \neq n}}[\log p(\boldsymbol{x}, \boldsymbol{z})] + \text{const}$

# Optimal Factors

$$\text{ELBO}(q_n) = -\text{KL}(q_n(\boldsymbol{z}_n) \| \exp(\hat{p}(\boldsymbol{x}, \boldsymbol{z}_n)))$$

- Maximizing the ELBO w.r.t. $q_n$ is equivalent to minimizing $\text{KL}(q_n(\boldsymbol{z}_n) \| \exp(\hat{p}(\boldsymbol{x}, \boldsymbol{z}_n)))$

$$\log q_n^*(\boldsymbol{z}_n) = \hat{p}(\boldsymbol{x}, \boldsymbol{z}_n) = \mathbb{E}_{q_{i \neq n}}[\log p(\boldsymbol{x}, \boldsymbol{z})] + \text{const}$$

- Get the optimal factor $q_n^*$ by

# Optimal Factors

$$\text{ELBO}(q_n) = -\text{KL}(q_n(\boldsymbol{z}_n) \| \exp(\hat{p}(\boldsymbol{x}, \boldsymbol{z}_n)))$$

- Maximizing the ELBO w.r.t. $q_n$ is equivalent to minimizing $\text{KL}(q_n(\boldsymbol{z}_n) \| \exp(\hat{p}(\boldsymbol{x}, \boldsymbol{z}_n)))$

$$\log q_n^*(\boldsymbol{z}_n) = \hat{p}(\boldsymbol{x}, \boldsymbol{z}_n) = \mathbb{E}_{q_{i \neq n}}[\log p(\boldsymbol{x}, \boldsymbol{z})] + \text{const}$$

- Get the optimal factor $q_n^*$ by
  1. Writing down the log-joint distribution of all latent and observed variables $\log p(\boldsymbol{x}, \boldsymbol{z})$

# Optimal Factors

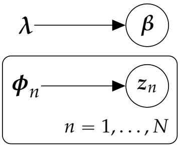
$$\text{ELBO}(q_n) = -\text{KL}(q_n(\boldsymbol{z}_n) \| \exp(\hat{p}(\boldsymbol{x}, \boldsymbol{z}_n)))$$

- Maximizing the ELBO w.r.t. $q_n$ is equivalent to minimizing $\text{KL}(q_n(\boldsymbol{z}_n) \| \exp(\hat{p}(\boldsymbol{x}, \boldsymbol{z}_n)))$

$$\log q_n^*(\boldsymbol{z}_n) = \hat{p}(\boldsymbol{x}, \boldsymbol{z}_n) = \mathbb{E}_{q_{i \neq n}}[\log p(\boldsymbol{x}, \boldsymbol{z})] + \text{const}$$

- Get the optimal factor $q_n^*$ by
  1. Writing down the log-joint distribution of all latent and observed variables $\log p(\boldsymbol{x}, \boldsymbol{z})$
  2. Computing the expectation w.r.t. all other random variables

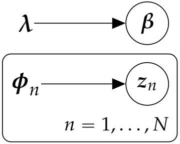
# Mean-Field Approximation for Conditionally Conjugate Models




- Optimal factors (see Bishop (2006) or Ghahramani & Beal (2001)):

$$q^*(\boldsymbol{\beta}|\boldsymbol{\lambda}) \propto \exp\left(\mathbb{E}_{\boldsymbol{z}}[\log p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\beta})]\right)$$
$$q^*(\boldsymbol{z}_n|\boldsymbol{\phi}_n) \propto \exp\left(\mathbb{E}_{\boldsymbol{\beta}}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n, \boldsymbol{\beta})]\right)$$

# Mean-Field Approximation for Conditionally Conjugate Models

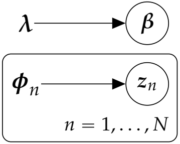



- Optimal factors (see Bishop (2006) or Ghahramani & Beal (2001)):

$$q^*(\boldsymbol{\beta}|\boldsymbol{\lambda}) \propto \exp\left(\mathbb{E}_{\boldsymbol{z}}[\log p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\beta})]\right)$$
$$q^*(\boldsymbol{z}_n|\boldsymbol{\phi}_n) \propto \exp\left(\mathbb{E}_{\boldsymbol{\beta}}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n, \boldsymbol{\beta})]\right)$$

- Update one term at a time ▸▸ Coordinate ascent

# Mean-Field Approximation for Conditionally Conjugate Models



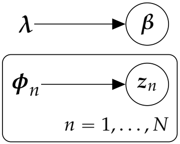

- Optimal factors (see Bishop (2006) or Ghahramani & Beal (2001)):

$$q^*(\boldsymbol{\beta}|\boldsymbol{\lambda}) \propto \exp\left(\mathbb{E}_{\boldsymbol{z}}[\log p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\beta})]\right)$$

$$q^*(\boldsymbol{z}_n|\boldsymbol{\phi}_n) \propto \exp\left(\mathbb{E}_{\boldsymbol{\beta}}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n, \boldsymbol{\beta})]\right)$$

- Update one term at a time ⏩ Coordinate ascent
- No closed-form solution (see EM algorithm)

# Mean-Field Approximation for Conditionally Conjugate Models




- Optimal factors (see Bishop (2006) or Ghahramani & Beal (2001)):

$$q^*(\boldsymbol{\beta}|\boldsymbol{\lambda}) \propto \exp\left(\mathbb{E}_{\boldsymbol{z}}[\log p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\beta})]\right)$$

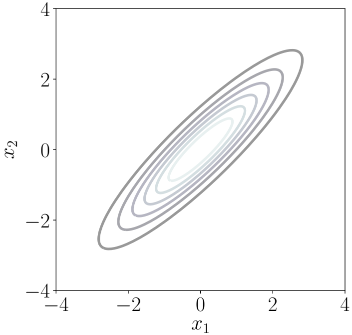
$$q^*(\boldsymbol{z}_n|\boldsymbol{\phi}_n) \propto \exp\left(\mathbb{E}_{\boldsymbol{\beta}}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n, \boldsymbol{\beta})]\right)$$

- Update one term at a time ⏩ Coordinate ascent
- No closed-form solution (see EM algorithm)
- Iteratively optimize each parameter until we reach a local optimum (convergence guaranteed)

# Mean-Field Approximation: Algorithm

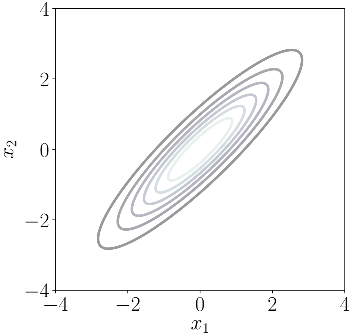
1. Input: data $\boldsymbol{x}$, model $p(\boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{x})$
2. Initialize global variational parameters $\boldsymbol{\lambda}$ randomly
3. While ELBO has not converged, repeat:
   - 3.1 For each data point $\boldsymbol{x}_n$
     - 3.1.1 Update local variational parameters $\boldsymbol{\phi}_n$
   - 3.2 Update global variational parameters $\boldsymbol{\lambda}$

# Mean-Field Approximation: Limitation



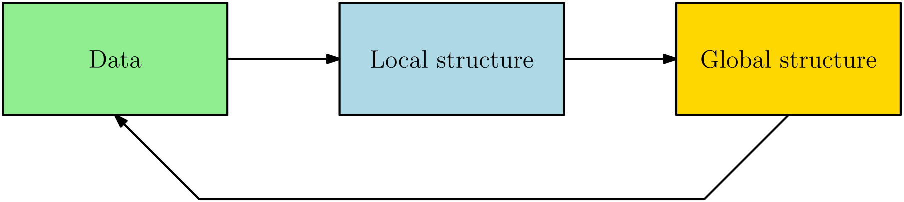

- Mean-field VI to approximate a correlated Gaussian with a factorized Gaussian

# Mean-Field Approximation: Limitation

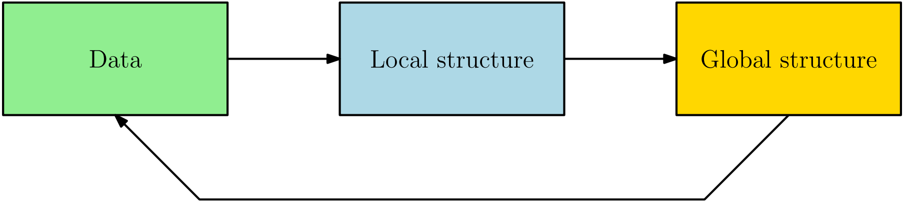



- Mean-field VI to approximate a correlated Gaussian with a factorized Gaussian
- Generally, mean-field VI tends to yield an approximation that is too compact ⏩ Need better classes of posterior approximations

# Classical Variational Inference: Limitation

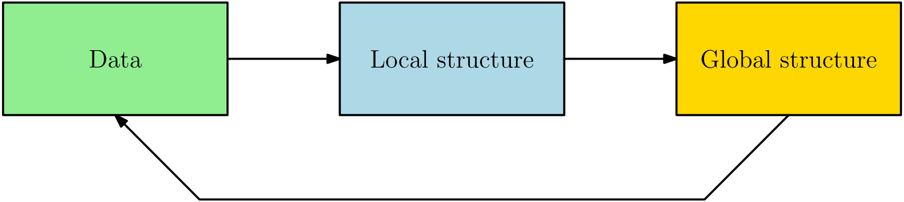


# Classical Variational Inference: Limitation




- Classical VI is inefficient: Need to crunch through the full dataset to update variational parameters
  **Can't handle massive data**

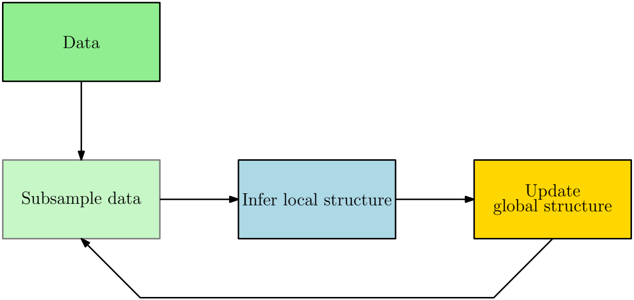
# Classical Variational Inference: Limitation




- Classical VI is inefficient: Need to crunch through the full dataset to update variational parameters
  **Can't handle massive data**
- **Stochastic variational inference** updates the global hidden structure once we have any update of the local structure
  ▸▸ Stochastic optimization

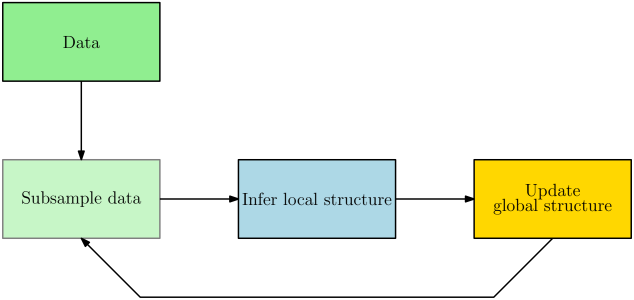
# Overview

# Stochastic Variational Inference



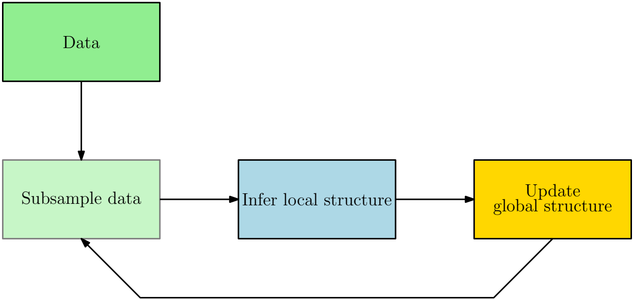

- Coordinate-ascent VI is inefficient: Need to look at the full dataset to update variational parameters

# Stochastic Variational Inference



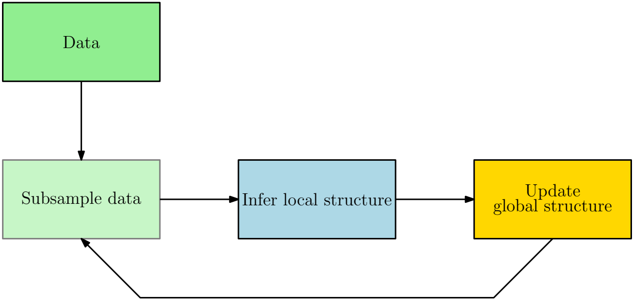

- Coordinate-ascent VI is inefficient: Need to look at the full dataset to update variational parameters
- Idea:

# Stochastic Variational Inference



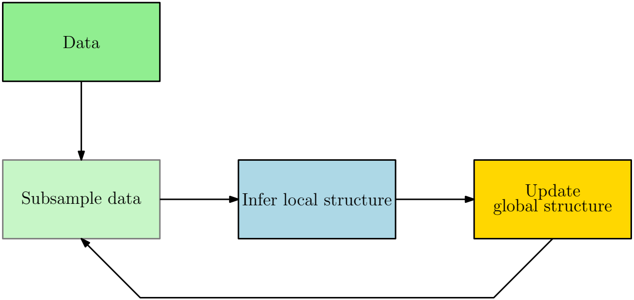

- Coordinate-ascent VI is inefficient: Need to look at the full dataset to update variational parameters
- Idea:
  - Do some local computations for each data point

# Stochastic Variational Inference




- Coordinate-ascent VI is inefficient: Need to look at the full dataset to update variational parameters
- Idea:
  - Do some local computations for each data point
  - Subsample data, infer local structure, update global structure

# Stochastic Variational Inference




- Coordinate-ascent VI is inefficient: Need to look at the full dataset to update variational parameters
- Idea:
  - Do some local computations for each data point
  - Subsample data, infer local structure, update global structure
- Key: Stochastic optimization

# Stochastic Optimization: Key Idea

- Replace exact gradient with cheaper (noisy) estimates (Robbins & Monro, 1951)
- This estimate could be based on a subset of the data (mini-batches)
- Guaranteed to converge to a local optimum
- Key driver of modern machine learning

# Noisy Updates of Variational Parameters

- With noisy gradients, update the variational parameters:

$$\boldsymbol{\nu}_{t+1} = \boldsymbol{\nu}_t + \rho_t \hat{\nabla}_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu})$$

# Noisy Updates of Variational Parameters

- With noisy gradients, update the variational parameters:

$$\boldsymbol{\nu}_{t+1} = \boldsymbol{\nu}_t + \rho_t \hat{\nabla}_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu})$$

- Requires unbiased gradients, i.e., on average the gradient points in the correct direction:

$$\mathbb{E}[\hat{\nabla}_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu})] = \nabla_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu})$$

# Noisy Updates of Variational Parameters

- With noisy gradients, update the variational parameters:

$$\boldsymbol{\nu}_{t+1} = \boldsymbol{\nu}_t + \rho_t \hat{\nabla}_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu})$$

- Requires unbiased gradients, i.e., on average the gradient points in the correct direction:

$$\mathbb{E}[\hat{\nabla}_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu})] = \nabla_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu})$$

- Some requirements on the step size parameter $\rho_t$
  ⏩ Convergence to local optimum

# Natural Gradient

$$\tilde{\nabla}_{\boldsymbol{\nu}}\mathcal{F}(\boldsymbol{\nu}) = \boldsymbol{F}^{-1}\nabla_{\boldsymbol{\nu}}\mathcal{F}(\boldsymbol{\nu}) \qquad \boldsymbol{F}: \text{ Fisher information matrix}$$

- Points in the direction of maximum change in distribution space, not in parameter space ⏩ We maximize the ELBO/minimize KL
- Invariant to parametrization of distribution (e.g., variance vs precision of a Gaussian)
- Scales each parameter individually

# Natural Gradient

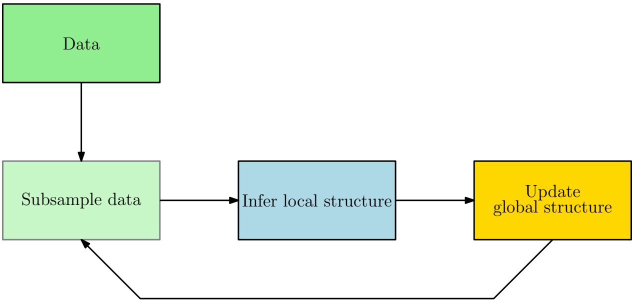
$$\tilde{\nabla}_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu}) = \boldsymbol{F}^{-1} \nabla_{\boldsymbol{\nu}} \mathcal{F}(\boldsymbol{\nu}) \qquad \boldsymbol{F}: \text{ Fisher information matrix}$$

- Points in the direction of maximum change in distribution space, not in parameter space ⏩ We maximize the ELBO/minimize KL
- Invariant to parametrization of distribution (e.g., variance vs precision of a Gaussian)
- Scales each parameter individually
- Natural gradient for conditionally conjugate models easy to compute (Hoffman et al., 2013)
- Noisy natural gradient (one estimate per data point)
- Unbiased
- Only depends on optimized parameters of a single data point
  ⏩ cheap to compute

# Algorithm

1. Input: data $\boldsymbol{x}$, model $p(\boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{x})$
2. Initialize global variational parameters $\boldsymbol{\lambda}$ randomly
3. Repeat
   3.1 Sample data point $\boldsymbol{x}_n$ uniformly at random
   3.2 Update local parameter $\boldsymbol{\phi}_n$
   3.3 Compute intermediate global parameter $\hat{\boldsymbol{\lambda}}$ based on noisy natural gradient
   3.4 Set global parameter

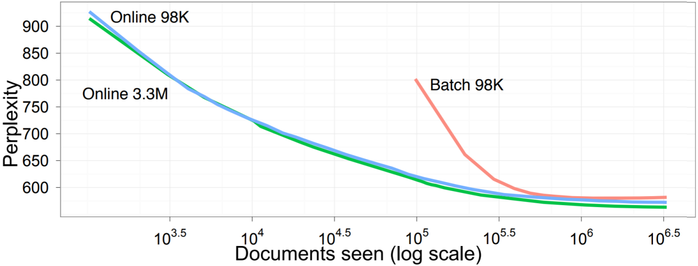
$$\boldsymbol{\lambda} \leftarrow (1 - \rho_t)\boldsymbol{\lambda} + \rho_t \hat{\boldsymbol{\lambda}}$$




- Look at a single data point in your dataset
- Infer local variational parameters
- Update global variational parameters using noisy natural gradient
- Repeat

▶▶ Simple way to scale variational inference to massive datasets

# Example: Online LDA (Hoffman et al., 2010)




| **Documents analyzed** | **2048** | **4096** | **8192** | **12288** | **16384** | **32768** | **49152** | **65536** |
|---|---|---|---|---|---|---|---|---|
| **Top eight words** | systems | systems | service | service | service | business | business | business |
| | road | health | systems | systems | companies | service | service | industry |
| | made | communication | health | companies | systems | companies | companies | service |
| | service | service | companies | business | business | industry | industry | companies |
| | announced | billion | market | company | company | company | services | services |
| | national | language | communication | billion | industry | management | company | company |
| | west | care | company | health | market | systems | management | management |
| | language | road | billion | industry | billion | services | public | public |

*From Hoffman et al. (2010)*

# Overview

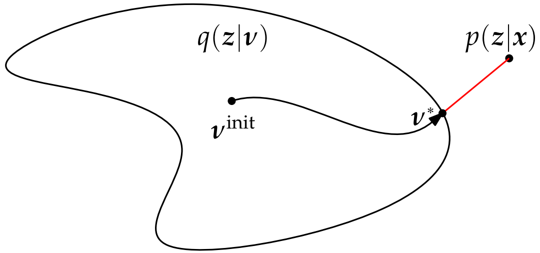



*Figure adopted from Blei et al.'s NIPS-2016 tutorial*

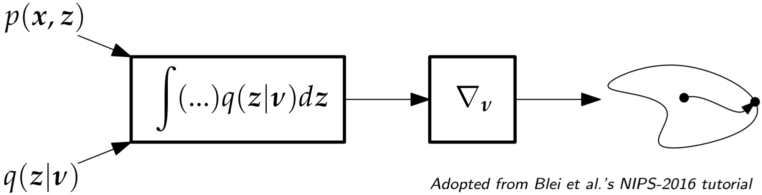
- Find approximation of a probability distribution (e.g., posterior) by optimization:
  1. Define an objective function
  2. Define a (parametrized) family of approximating distributions $q_{\boldsymbol{\nu}}$
  3. Optimize objective function w.r.t. variational parameters $\boldsymbol{\nu}$
- Inference ⏩ Optimization

# Roadmap II

1. Limits of Classical Variational Inference
2. Black-Box Variational Inference
3. Computing Gradients of Expectations

# Overview

# Variational Inference: General Recipe



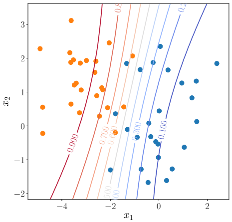

*Adopted from Blei et al.'s NIPS-2016 tutorial*

- Specify model $p(\boldsymbol{x}, \boldsymbol{z})$ and approximation $q(\boldsymbol{z}|\boldsymbol{\nu})$
- Objective $\mathcal{F}(\boldsymbol{\nu}) = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})]$
- Compute expectation
- Compute gradient
- Optimize with gradient descent

This recipe is fairly generic.

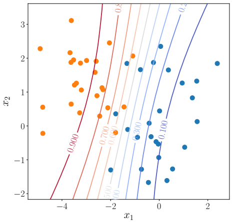
# Example: Bayesian Logistic Regression

- Binary classification
- Inputs $x \in \mathbb{R}$, labels $y \in \{0, 1\}$
- Model parameter $z$ (normally denoted by $\theta$)



# Example: Bayesian Logistic Regression

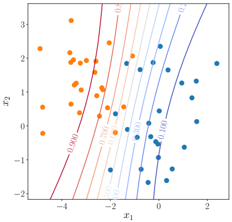
- Binary classification
- Inputs $x \in \mathbb{R}$, labels $y \in \{0, 1\}$
- Model parameter $z$ (normally denoted by $\theta$)



$$\text{Prior on model parameter: } p(z) = \mathcal{N}\big(0,\, 1\big)$$
$$\text{Likelihood: } p(y_n | x_n, z) = \text{Ber}(\sigma(z x_n))$$

# Example: Bayesian Logistic Regression

- Binary classification
- Inputs $x \in \mathbb{R}$, labels $y \in \{0, 1\}$
- Model parameter $z$ (normally denoted by $\theta$)




$$\text{Prior on model parameter: } p(z) = \mathcal{N}\big(0,\, 1\big)$$
$$\text{Likelihood: } p(y_n|x_n, z) = \text{Ber}(\sigma(zx_n))$$

- Assume we have a single data point $(x, y)$
- Goal: Approximate the intractable posterior distribution $p(z|x, y)$ using variational inference

# Example: Bayesian Logistic Regression (2)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}(\mu, \sigma^2)$ ▶▶ $\boldsymbol{\nu} =$

# Example: Bayesian Logistic Regression (2)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}\left(\mu, \sigma^2\right)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$

# Example: Bayesian Logistic Regression (2)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}\left(\mu,\, \sigma^2\right)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$
- Objective function: ELBO $\mathcal{F}(\boldsymbol{\nu})$

$$\mathcal{F}(m, \sigma^2) = \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y|x,z)]$$

# Example: Bayesian Logistic Regression (2)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}\big(\mu,\, \sigma^2\big)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$
- Objective function: ELBO $\mathcal{F}(\boldsymbol{\nu})$

$$\begin{aligned}\mathcal{F}(m, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y|x, z)] \\ &= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y|x, z)] + \mathrm{c}\end{aligned}$$

# Example: Bayesian Logistic Regression (2)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}\big(\mu,\, \sigma^2\big)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$
- Objective function: ELBO $\mathcal{F}(\boldsymbol{\nu})$

$$\begin{aligned}\mathcal{F}(m, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y|x, z)] \\ &= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y|x,z)] + \mathrm{c}\end{aligned}$$

$$\mathbb{E}_q[\log p(y|x,z)] =$$

# Example: Bayesian Logistic Regression (2)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}(\mu, \sigma^2)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$
- Objective function: ELBO $\mathcal{F}(\boldsymbol{\nu})$

$$\begin{aligned}\mathcal{F}(m, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y|x,z)] \\ &= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log\sigma^2 + \mathbb{E}_q[\log p(y|x,z)] + \mathrm{c} \\ \mathbb{E}_q[\log p(y|x,z)] &= \mathbb{E}_q[y \log\sigma(xz) + (1-y)\log(1-\sigma(xz))]\end{aligned}$$

# Example: Bayesian Logistic Regression (2)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}\left(\mu,\, \sigma^2\right)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$
- Objective function: ELBO $\mathcal{F}(\boldsymbol{\nu})$

$$\begin{aligned}\mathcal{F}(m, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y|x, z)] \\ &= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y|x, z)] + \mathrm{c} \\ \mathbb{E}_q[\log p(y|x, z)] &= \mathbb{E}_q[y \log \sigma(xz) + (1 - y)\log(1 - \sigma(xz))] \\ &= \mathbb{E}_q[yxz] - \mathbb{E}_q[y \log(1 + \exp(xz))] \\ &\quad + \mathbb{E}_q\left[(1 - y)\log\left(1 - \frac{\exp(xz)}{1 + \exp(xz)}\right)\right]\end{aligned}$$

with

$$\sigma(xz) = \frac{\exp(xz)}{1 + \exp(xz)}$$

# Computing the Expected Log-Likelihood

$$\mathbb{E}_q[\log p(y|x,z)] = \mathbb{E}_q[yxz] - \mathbb{E}_q[y \log(1+\exp(xz))] + \mathbb{E}_q\left[(1-y)\log\left(1 - \frac{\exp(xz)}{1+\exp(xz)}\right)\right]$$

# Computing the Expected Log-Likelihood

$$
\begin{aligned}
\mathbb{E}_q[\log p(y|x,z)] &= \mathbb{E}_q[yxz] - \mathbb{E}_q[y\log(1+\exp(xz))] \\
&\quad + \mathbb{E}_q[(1-y)\log\left(1 - \frac{\exp(xz)}{1+\exp(xz)}\right)] \\
&= yx\mu - \mathbb{E}_q[y\log(1+\exp(xz))] \\
&\quad + \mathbb{E}_q[(1-y)\log\left(\frac{1}{1+\exp(xz)}\right)]
\end{aligned}
$$

# Computing the Expected Log-Likelihood

$$
\begin{aligned}
\mathbb{E}_q[\log p(y|x,z)] &= \mathbb{E}_q[yxz] - \mathbb{E}_q[y\log(1+\exp(xz))] \\
&\quad + \mathbb{E}_q[(1-y)\log\left(1 - \frac{\exp(xz)}{1+\exp(xz)}\right)] \\
&= yx\mu - \mathbb{E}_q[y\log(1+\exp(xz))] \\
&\quad + \mathbb{E}_q[(1-y)\log\left(\frac{1}{1+\exp(xz)}\right)] \\
&= yx\mu - \mathbb{E}_q[y\log(1+\exp(xz))] \\
&\quad - \mathbb{E}_q[\log(1+\exp(xz))] + \mathbb{E}_q[y\log(1+\exp(xz))]
\end{aligned}
$$

# Computing the Expected Log-Likelihood

$$
\begin{aligned}
\mathbb{E}_q[\log p(y|x,z)] &= \mathbb{E}_q[yxz] - \mathbb{E}_q[y\log(1+\exp(xz))] \\
&\quad + \mathbb{E}_q[(1-y)\log\left(1-\frac{\exp(xz)}{1+\exp(xz)}\right)] \\
&= yx\mu - \mathbb{E}_q[y\log(1+\exp(xz))] \\
&\quad + \mathbb{E}_q[(1-y)\log\left(\frac{1}{1+\exp(xz)}\right)] \\
&= yx\mu - \mathbb{E}_q[y\log(1+\exp(xz))] \\
&\quad - \mathbb{E}_q[\log(1+\exp(xz))] + \mathbb{E}_q[y\log(1+\exp(xz))] \\
&= yx\mu - \mathbb{E}_q[\log(1+\exp(xz))]
\end{aligned}
$$

# Example: Bayesian Logistic Regression (ctd.)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}\big(\mu,\, \sigma^2\big)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$
- Objective function: ELBO $\mathcal{F}(\boldsymbol{\nu})$

$$
\begin{aligned}
\mathcal{F}(\mu, \sigma^2) &= \mathbb{E}_q[\log p(z) + \log p(y|x, z) - \log q(z)] \\
&= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y|x, z)] + \mathrm{c} \\
&= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log \sigma^2 + yx\mu - \mathbb{E}_q[\log(1 + \exp(xz))]
\end{aligned}
$$

# Example: Bayesian Logistic Regression (ctd.)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}\left(\mu,\, \sigma^2\right)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$
- Objective function: ELBO $\mathcal{F}(\boldsymbol{\nu})$

$$\begin{aligned}\mathcal{F}(\mu, \sigma^2) &= \mathbb{E}_q[\log p(z) + \log p(y|x,z) - \log q(z)] \\ &= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log\sigma^2 + \mathbb{E}_q[\log p(y|x,z)] + \mathrm{c} \\ &= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log\sigma^2 + yx\mu - \mathbb{E}_q[\log(1 + \exp(xz))]\end{aligned}$$

- **Expectation cannot be computed in closed form**
- Pushing gradients through Monte Carlo estimates is very hard.

# Example: Bayesian Logistic Regression (ctd.)

- Choose Gaussian variational approximation:
  $q(z|\boldsymbol{\nu}) = \mathcal{N}\left(\mu,\, \sigma^2\right)$ ⏩ $\boldsymbol{\nu} = \{\mu, \sigma^2\}$
- Objective function: ELBO $\mathcal{F}(\boldsymbol{\nu})$

$$\begin{aligned}\mathcal{F}(\mu, \sigma^2) &= \mathbb{E}_q[\log p(z) + \log p(y|x, z) - \log q(z)] \\ &= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log\sigma^2 + \mathbb{E}_q[\log p(y|x, z)] + \mathrm{c} \\ &= -\tfrac{1}{2}(\mu^2 + \sigma^2) + \tfrac{1}{2}\log\sigma^2 + yx\mu - \mathbb{E}_q[\log(1 + \exp(xz))]\end{aligned}$$

- **Expectation cannot be computed in closed form**
- Pushing gradients through Monte Carlo estimates is very hard.
- Option: Lower-bound this expectation; but this is model specific.

# Non-Conjugate Models

- Nonlinear time series models
- Deep latent Gaussian models
- Attention models (e.g., DRAW)
- Generalized linear models (e.g., logistic regression)
- Bayesian neural networks
- ...

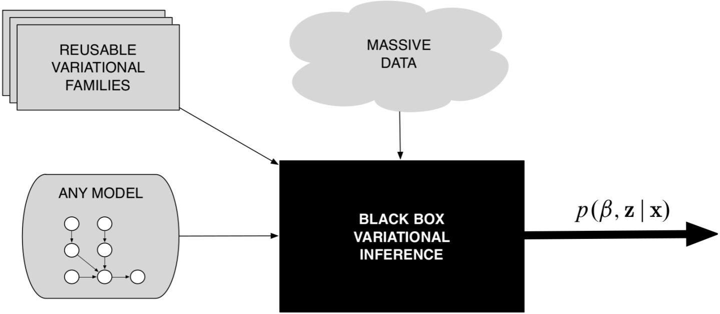
# Non-Conjugate Models

- Nonlinear time series models
- Deep latent Gaussian models
- Attention models (e.g., DRAW)
- Generalized linear models (e.g., logistic regression)
- Bayesian neural networks
- ...

There are many interesting non-conjugate models
▶▶ Look for a solution that is not model specific
▶▶ **Black-Box Variational Inference**

# Overview

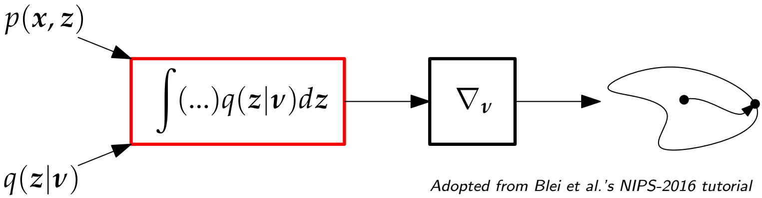
# Black-Box Variational Inference




*From Blei et al.'s NIPS-2016 tutorial*

- Any model
- Massive data
- Some general assumptions on the approximating family

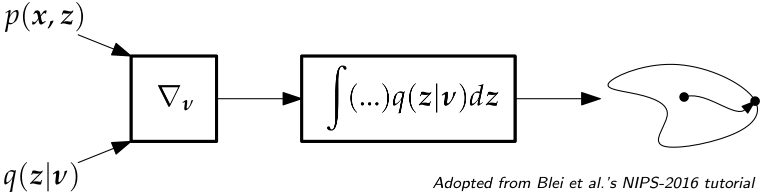
# Computational Challenge of Classical VI




*Adopted from Blei et al.'s NIPS-2016 tutorial*

- Integral computation, which makes the ELBO explicitly a function of the variational parameters
- Integral cannot be computed for non-conjugate models
  ⏩ Gradient computation difficult

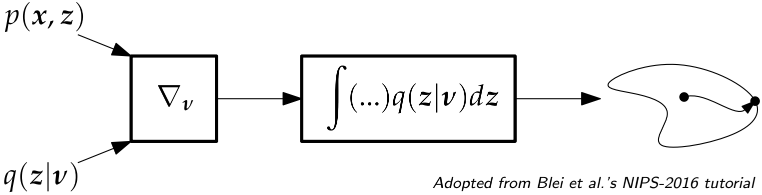
# Approach




*Adopted from Blei et al.'s NIPS-2016 tutorial*

- Switch order of integration (compute expectations) and differentiation

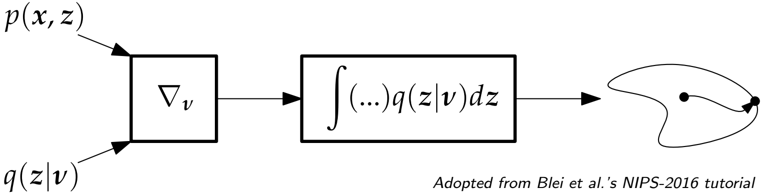
# Approach




*Adopted from Blei et al.'s NIPS-2016 tutorial*

- Switch order of integration (compute expectations) and differentiation
- Approximate the expectation after having taken the gradient
  ▶▶ Monte Carlo estimator (ideally with low variance)

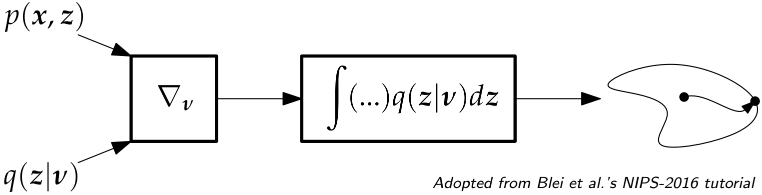
# Approach




*Adopted from Blei et al.'s NIPS-2016 tutorial*

- Switch order of integration (compute expectations) and differentiation
- Approximate the expectation after having taken the gradient
  ▸▸ Monte Carlo estimator (ideally with low variance)
- Stochastic optimization

# Approach




*Adopted from Blei et al.'s NIPS-2016 tutorial*

- Switch order of integration (compute expectations) and differentiation
- Approximate the expectation after having taken the gradient
  ⏩ Monte Carlo estimator (ideally with low variance)
- Stochastic optimization

⏩ Require a general way to compute gradients of expectations

# Re-Writing the ELBO

$$\text{ELBO} = \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \text{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}))$$

# Re-Writing the ELBO

$$\begin{aligned}\text{ELBO} &= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \text{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}))\\ &= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \mathbb{E}_q[\log p(\boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})]\end{aligned}$$

# Re-Writing the ELBO

$$\begin{aligned}\text{ELBO} &= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \text{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z}))\\ &= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \mathbb{E}_q[\log p(\boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})]\\ &= \mathbb{E}_q[\underbrace{\log p(\boldsymbol{x},\boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})}_{=:g(\boldsymbol{z},\boldsymbol{\nu})}]\end{aligned}$$

# Re-Writing the ELBO

$$
\begin{aligned}
\text{ELBO} &= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] - \text{KL}(q(\boldsymbol{z}|\boldsymbol{\nu})\|p(\boldsymbol{z})) \\
&= \mathbb{E}_q[\log p(\boldsymbol{x}|\boldsymbol{z})] + \mathbb{E}_q[\log p(\boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})] \\
&= \mathbb{E}_q[\underbrace{\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})}_{=:g(\boldsymbol{z},\boldsymbol{\nu})}]
\end{aligned}
$$

## Evidence Lower Bound

$$
\text{ELBO} = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] = \int g(\boldsymbol{z}, \boldsymbol{\nu}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z}
$$

$$
g(\boldsymbol{z}, \boldsymbol{\nu}) := \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})
$$

# Overview

Introduction and Background

Key Idea

Optimization Objective

Conditionally Conjugate Models

Mean-Field Variational Inference

Stochastic Variational Inference

Limits of Classical Variational Inference

Black-Box Variational Inference

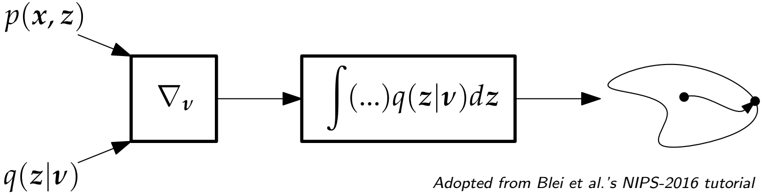
Computing Gradients of Expectations

- Score Function Gradients
- Pathwise Gradients

Amortized Inference

Richer Posterior Approximations

# Approach




*Adopted from Blei et al.'s NIPS-2016 tutorial*

- **Switch order of integration (compute expectations) and differentiation**
- Simplify the expectation after having taken the gradient

# Log-Derivative Trick

**Log-Derivative Trick**

$$\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) =$$

# Log-Derivative Trick

## Log-Derivative Trick

$$\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) = \frac{\nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})}{q(\boldsymbol{z}|\boldsymbol{\nu})}$$

$$\iff \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

# Log-Derivative Trick

**Log-Derivative Trick**

$$\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) = \frac{\nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})}{q(\boldsymbol{z}|\boldsymbol{\nu})}$$

$$\iff \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Therefore:

$$\int \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) f(\boldsymbol{z}) d\boldsymbol{z} = \int q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) f(\boldsymbol{z}) d\boldsymbol{z}$$

$$= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) f(\boldsymbol{z})]$$

# Log-Derivative Trick

## Log-Derivative Trick

$$\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) = \frac{\nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})}{q(\boldsymbol{z}|\boldsymbol{\nu})}$$

$$\iff \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) = q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Therefore:

$$\int \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) f(\boldsymbol{z}) d\boldsymbol{z} = \int q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) f(\boldsymbol{z}) d\boldsymbol{z}$$
$$= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) f(\boldsymbol{z})]$$

- If we can sample from $q$, this expectation can be evaluated easily (Monte Carlo estimation)

# Gradients of Expectations: Approach 1

$$\text{ELBO} = \mathcal{F}(\boldsymbol{\nu}) = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})]\,, \quad g(\boldsymbol{z}, \boldsymbol{\nu}) = \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Need gradient of ELBO w.r.t. variational parameters $\boldsymbol{\nu}$

# Gradients of Expectations: Approach 1

$$\text{ELBO} = \mathcal{F}(\boldsymbol{\nu}) = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})], \quad g(\boldsymbol{z}, \boldsymbol{\nu}) = \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Need gradient of ELBO w.r.t. variational parameters $\boldsymbol{\nu}$

$$\nabla_{\boldsymbol{\nu}} \mathcal{F} = \nabla_{\boldsymbol{\nu}} \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})]$$

$$= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu}) + \nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu})]$$

# Gradients of Expectations: Approach 1

$$\text{ELBO} = \mathcal{F}(\boldsymbol{\nu}) = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})], \quad g(\boldsymbol{z}, \boldsymbol{\nu}) = \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Need gradient of ELBO w.r.t. variational parameters $\boldsymbol{\nu}$

$$\nabla_{\boldsymbol{\nu}} \mathcal{F} = \nabla_{\boldsymbol{\nu}} \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] = \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z}, \boldsymbol{\nu}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z}$$

$$= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu}) + \nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu})]$$

# Gradients of Expectations: Approach 1

$$\text{ELBO} = \mathcal{F}(\boldsymbol{\nu}) = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] \,, \quad g(\boldsymbol{z}, \boldsymbol{\nu}) = \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Need gradient of ELBO w.r.t. variational parameters $\boldsymbol{\nu}$

$$\nabla_{\boldsymbol{\nu}} \mathcal{F} = \nabla_{\boldsymbol{\nu}} \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] = \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z}, \boldsymbol{\nu}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z}$$

$$= \int \big(\nabla_{\boldsymbol{\nu}} g(\boldsymbol{\nu}, \boldsymbol{z})\big) q(\boldsymbol{z}|\boldsymbol{\nu}) + g(\boldsymbol{\nu}, \boldsymbol{z}) \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z}$$

product rule

$$= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu}) + \nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu})]$$

# Gradients of Expectations: Approach 1

$$\text{ELBO} = \mathcal{F}(\boldsymbol{\nu}) = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] , \quad g(\boldsymbol{z}, \boldsymbol{\nu}) = \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Need gradient of ELBO w.r.t. variational parameters $\boldsymbol{\nu}$

$$\begin{aligned}
\nabla_{\boldsymbol{\nu}} \mathcal{F} = \nabla_{\boldsymbol{\nu}} \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] &= \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z}, \boldsymbol{\nu}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} \\
&= \int \big(\nabla_{\boldsymbol{\nu}} g(\boldsymbol{\nu}, \boldsymbol{z})\big) q(\boldsymbol{z}|\boldsymbol{\nu}) + g(\boldsymbol{\nu}, \boldsymbol{z}) \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} && \boxed{\text{product rule}} \\
&= \int q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu}) + q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu}) d\boldsymbol{z} && \boxed{\text{log-deriv. trick}} \\
&= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu}) + \nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu})]
\end{aligned}$$

# Gradients of Expectations: Approach 1

$$\text{ELBO} = \mathcal{F}(\boldsymbol{\nu}) = \mathbb{E}_q[g(\boldsymbol{z},\boldsymbol{\nu})]\,, \quad g(\boldsymbol{z},\boldsymbol{\nu}) = \log p(\boldsymbol{x},\boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Need gradient of ELBO w.r.t. variational parameters $\boldsymbol{\nu}$

$$\nabla_{\boldsymbol{\nu}}\mathcal{F} = \nabla_{\boldsymbol{\nu}}\mathbb{E}_q[g(\boldsymbol{z},\boldsymbol{\nu})] = \nabla_{\boldsymbol{\nu}}\int g(\boldsymbol{z},\boldsymbol{\nu})q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z}$$

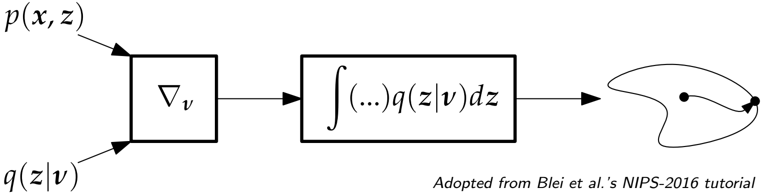
$$= \int (\nabla_{\boldsymbol{\nu}} g(\boldsymbol{\nu},\boldsymbol{z}))q(\boldsymbol{z}|\boldsymbol{\nu}) + g(\boldsymbol{\nu},\boldsymbol{z})\nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z} \qquad \boxed{\text{product rule}}$$

$$= \int q(\boldsymbol{z}|\boldsymbol{\nu})\nabla_{\boldsymbol{\nu}} g(\boldsymbol{z},\boldsymbol{\nu}) + q(\boldsymbol{z}|\boldsymbol{\nu})\nabla_{\boldsymbol{\nu}}\log q(\boldsymbol{z}|\boldsymbol{\nu})g(\boldsymbol{z},\boldsymbol{\nu})d\boldsymbol{z} \qquad \boxed{\text{log-deriv. trick}}$$

$$= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}}\log q(\boldsymbol{z}|\boldsymbol{\nu})g(\boldsymbol{z},\boldsymbol{\nu}) + \nabla_{\boldsymbol{\nu}} g(\boldsymbol{z},\boldsymbol{\nu})]$$

- We successfully swapped gradient and expectation
- $q$ known
  - ▶▶ Sample from $q$ and use Monte Carlo estimation

# Approach




*Adopted from Blei et al.'s NIPS-2016 tutorial*

- Swap order of integration (compute expectations) and differentiation
- **Simplify the expectation after having taken the gradient**

# Score Function Gradient Estimator of the ELBO

# Simplifying the Gradient

$$\nabla_{\boldsymbol{\nu}}\text{ELBO} = \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})g(\boldsymbol{z},\boldsymbol{\nu}) + \nabla_{\boldsymbol{\nu}} g(\boldsymbol{z},\boldsymbol{\nu})]$$

- Let's simplify this gradient ⏩ Score function

# Score Function

- Score function: Derivative of a log-likelihood with respect to the parameter vector $\boldsymbol{\nu}$:

# Score Function

- Score function: Derivative of a log-likelihood with respect to the parameter vector $\boldsymbol{\nu}$:

**Score Function**

$$\text{score} = \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) = \frac{1}{q(\boldsymbol{z}|\boldsymbol{\nu})} \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})$$

# Score Function

- Score function: Derivative of a log-likelihood with respect to the parameter vector $\boldsymbol{\nu}$:

**Score Function**

$$\text{score} = \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) = \frac{1}{q(\boldsymbol{z}|\boldsymbol{\nu})} \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Measures the sensitivity of the log-likelihood w.r.t. $\boldsymbol{\nu}$

# Score Function

- Score function: Derivative of a log-likelihood with respect to the parameter vector $\boldsymbol{\nu}$:

**Score Function**

$$\text{score} = \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) = \frac{1}{q(\boldsymbol{z}|\boldsymbol{\nu})} \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Measures the sensitivity of the log-likelihood w.r.t. $\boldsymbol{\nu}$
- Central to maximum likelihood estimation

# Score Function (2)

$$\text{score} = \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) = \frac{1}{q(\boldsymbol{z}|\boldsymbol{\nu})} \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Important property:

$$\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[\text{score}] =$$

# Score Function (2)

$$\text{score} = \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) = \frac{1}{q(\boldsymbol{z}|\boldsymbol{\nu})} \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Important property:

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[\text{score}] &= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}\left[\frac{1}{q(\boldsymbol{z}|\boldsymbol{\nu})} \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu})\right] \\
&= \int \frac{1}{q(\boldsymbol{z}|\boldsymbol{\nu})} q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} \\
&= \int \nabla_{\boldsymbol{\nu}} q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} = \nabla_{\boldsymbol{\nu}} \int q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} = \nabla_{\boldsymbol{\nu}} 1 = 0
\end{aligned}$$

  ⏩ Mean of the score function is 0

- Variance of the score: Fisher information ⏩ Natural gradients

# Score Function Gradient Estimator

$$\text{ELBO} = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Gradient of ELBO:

$$\nabla_{\boldsymbol{\nu}} \text{ELBO} =$$

# Score Function Gradient Estimator

$$\text{ELBO} = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Gradient of ELBO:

$$\nabla_{\boldsymbol{\nu}}\text{ELBO} = \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] + \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu})]$$

# Score Function Gradient Estimator

$$\text{ELBO} = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Gradient of ELBO:

$$\begin{aligned}\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] + \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu})] \\ &= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] \\ &\quad + \mathbb{E}_q[\underbrace{\nabla_{\boldsymbol{\nu}} \log p(\boldsymbol{x}, \boldsymbol{z})}_{=0} - \underbrace{\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})}_{\text{score}}]\end{aligned}$$

# Score Function Gradient Estimator

$$\text{ELBO} = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Gradient of ELBO:

$$\begin{aligned}
\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] + \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu})] \\
&= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] \\
&\quad + \mathbb{E}_q[\underbrace{\nabla_{\boldsymbol{\nu}} \log p(\boldsymbol{x}, \boldsymbol{z})}_{=0} - \underbrace{\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})}_{\text{score}}]
\end{aligned}$$

- Exploit that the mean of the score function is 0. Then:

$$\begin{aligned}
\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] \\
&= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu}))]
\end{aligned}$$

# Score Function Gradient Estimator

$$\text{ELBO} = \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})]$$

- Gradient of ELBO:

$$\begin{aligned}\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] + \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} g(\boldsymbol{z}, \boldsymbol{\nu})] \\ &= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] \\ &\quad + \mathbb{E}_q[\underbrace{\nabla_{\boldsymbol{\nu}} \log p(\boldsymbol{x}, \boldsymbol{z})}_{=0} - \underbrace{\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})}_{\text{score}}]\end{aligned}$$

- Exploit that the mean of the score function is 0. Then:

$$\begin{aligned}\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) g(\boldsymbol{z}, \boldsymbol{\nu})] \\ &= \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu}))]\end{aligned}$$

- Likelihood ratio gradient (Glynn, 1990)
- REINFORCE gradient (Williams, 1992)

# Using Noisy Stochastic Gradients

- Gradient of the ELBO

$$\nabla_{\boldsymbol{\nu}}\text{ELBO} = \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu}))]$$

is an expectation

# Using Noisy Stochastic Gradients

- Gradient of the ELBO

$$\nabla_{\boldsymbol{\nu}} \text{ELBO} = \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu}))]$$

  is an expectation
- Require that $q(\boldsymbol{z}|\boldsymbol{\nu})$ is differentiable w.r.t. $\boldsymbol{\nu}$

# Using Noisy Stochastic Gradients

- Gradient of the ELBO

$$\nabla_{\boldsymbol{\nu}}\text{ELBO} = \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu}))]$$

  is an expectation
- Require that $q(\boldsymbol{z}|\boldsymbol{\nu})$ is differentiable w.r.t. $\boldsymbol{\nu}$
- Get noisy unbiased gradients using Monte Carlo by sampling from $q$:

$$\frac{1}{S}\sum_{s=1}^{S} \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}^{(s)}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}^{(s)}) - \log q(\boldsymbol{z}^{(s)}|\boldsymbol{\nu}))\,, \quad \boldsymbol{z}^{(s)} \sim q(\boldsymbol{z}|\boldsymbol{\nu})$$

# Using Noisy Stochastic Gradients

- Gradient of the ELBO

$$\nabla_{\boldsymbol{\nu}}\text{ELBO} = \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu}))]$$

  is an expectation
- Require that $q(\boldsymbol{z}|\boldsymbol{\nu})$ is differentiable w.r.t. $\boldsymbol{\nu}$
- Get noisy unbiased gradients using Monte Carlo by sampling from $q$:

$$\frac{1}{S}\sum_{s=1}^{S} \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}^{(s)}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}^{(s)}) - \log q(\boldsymbol{z}^{(s)}|\boldsymbol{\nu}))\,, \quad \boldsymbol{z}^{(s)} \sim q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Sampling from $q$ is easy (we choose $q$)

# Using Noisy Stochastic Gradients

- Gradient of the ELBO

$$\nabla_{\boldsymbol{\nu}} \text{ELBO} = \mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu}))]$$

  is an expectation

- Require that $q(\boldsymbol{z}|\boldsymbol{\nu})$ is differentiable w.r.t. $\boldsymbol{\nu}$
- Get noisy unbiased gradients using Monte Carlo by sampling from $q$:

$$\frac{1}{S}\sum_{s=1}^{S} \nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}^{(s)}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}^{(s)}) - \log q(\boldsymbol{z}^{(s)}|\boldsymbol{\nu}))\,, \quad \boldsymbol{z}^{(s)} \sim q(\boldsymbol{z}|\boldsymbol{\nu})$$

- Sampling from $q$ is easy (we choose $q$)
- Use this within SVI to converge to a local optimum

# BBVI: Algorithm

1. Input: model $p(\boldsymbol{x}, \boldsymbol{z})$, variational approximation $q(\boldsymbol{z}|\boldsymbol{\nu})$
2. Repeat
   2.1 Draw $S$ samples $\boldsymbol{z}^{(s)} \sim q(\boldsymbol{z}|\boldsymbol{\nu})$
   2.2 Update variational parameters

$$\boldsymbol{\nu}_{t+1} = \boldsymbol{\nu}_t + \rho_t \underbrace{\frac{1}{S}\sum_{s=1}^{S} \nabla_{\nu} \log q(\boldsymbol{z}^{(s)}|\boldsymbol{\nu})(\log p(\boldsymbol{x}, \boldsymbol{z}^{(s)}) - \log q(\boldsymbol{z}^{(s)}|\boldsymbol{\nu}))}_{\text{MC estimate of the score-function gradient of the ELBO}}$$

   2.3 $t = t + 1$

# Requirements for Inference

- Computing the noisy gradient of the ELBO requires:
  - Sampling from $q$. We choose $q$ so that this is possible.
  - Evaluate the score function $\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})$
  - Evaluate $\log q(\boldsymbol{z}|\boldsymbol{\nu})$ and $\log p(\boldsymbol{x}, \boldsymbol{z}) = \log p(\boldsymbol{z}) + \log p(\boldsymbol{x}|\boldsymbol{z})$

▶▶ No model-specific computations for optimization (computations are only specific to the choice of the variational approximation)

# Issue: Variance of the Gradients

- Stochastic optimization ⏩ **Gradients are noisy (high variance)**
- The noisier the gradients, the slower the convergence
- Possible solutions:
  - Control variates (with the score function as control variate)
  - Rao-Blackwellization
  - Importance sampling

# Non-Conjugate Models

- Nonlinear time series models
- Deep latent Gaussian models
- Attention models (e.g., DRAW)
- Generalized linear models (e.g., logistic regression)
- Bayesian neural networks
- ...

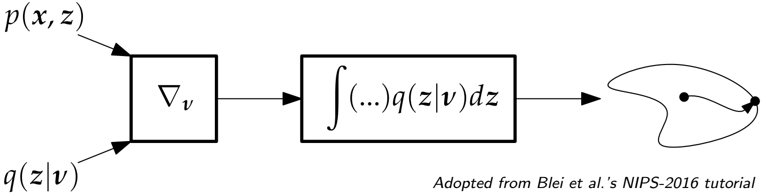
BBVI allows us to design models $p(\boldsymbol{x}, \boldsymbol{z})$ based on the data, and not on the inference we can do

# Assumptions

- Score-function gradient estimator only requires general assumptions
- Noisy gradients are a problem
- Address this issue by making some additional assumptions (not too strict)
  ▸▸ Pathwise gradient estimators

## Pathwise Gradient Estimators of the ELBO

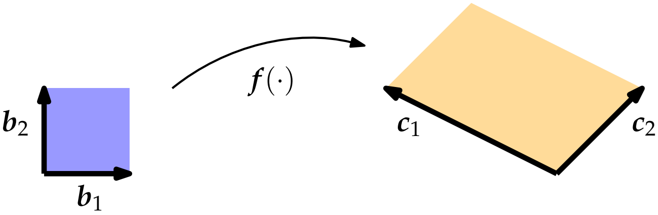
# Approach




*Adopted from Blei et al.'s NIPS-2016 tutorial*

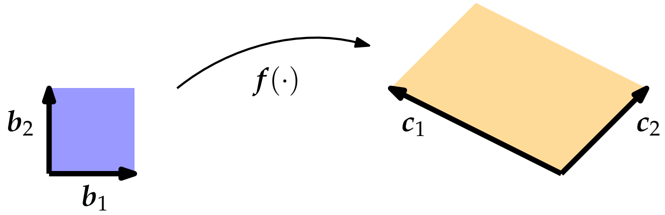
- **Switch order of integration (compute expectations) and differentiation**
- Approximate the expectation after having taken the gradient

# Change of Variables




- Use function $f : \mathcal{X} \to \mathcal{Y}, \quad x \mapsto f(x) = y$
- Change of area/volume:

# Change of Variables




- Use function $f : \mathcal{X} \to \mathcal{Y}, \quad x \mapsto f(x) = y$
- Change of area/volume:

$$\left|\int_{\mathcal{Y}} y d y\right| = \left|\frac{\mathrm{d} f}{\mathrm{d} x}\right| \left|\int_{\mathcal{X}} x d x\right|$$

# Reparametrization Trick

## Reparametrization Trick

Base distribution $p(\boldsymbol{\epsilon})$ and a deterministic transformation $\boldsymbol{z} = t(\boldsymbol{\epsilon}, \boldsymbol{\nu})$ so that $\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{\nu})$. Then:

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]$$

▶▶ Expectation taken w.r.t. base distribution

# Reparametrization Trick

**Reparametrization Trick**

Base distribution $p(\boldsymbol{\epsilon})$ and a deterministic transformation $\boldsymbol{z} = t(\boldsymbol{\epsilon}, \boldsymbol{\nu})$ so that $\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{\nu})$. Then:

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]$$

▶▶ Expectation taken w.r.t. base distribution

- Key idea: change of variables using a deterministic transformation

# Reparametrization Trick (2)

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]$$

# Reparametrization Trick (2)

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]$$

- Change of variables ⏩ Probability mass contained in a differential area must be invariant under change of variables:

$$|q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z}| = |p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}|$$

# Reparametrization Trick (2)

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]$$

- Change of variables ⏩ Probability mass contained in a differential area must be invariant under change of variables:

$$|q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z}| = |p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}|$$

- This implies

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] =$$

# Reparametrization Trick (2)

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]$$

- Change of variables ⏩ Probability mass contained in a differential area must be invariant under change of variables:

$$|q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z}| = |p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}|$$

- This implies

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \nabla_{\boldsymbol{\nu}} \int f(\boldsymbol{z}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} = \nabla_{\boldsymbol{\nu}} \int f(\boldsymbol{z}(\boldsymbol{\epsilon})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}$$

# Reparametrization Trick (2)

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]$$

- Change of variables ⏩ Probability mass contained in a differential area must be invariant under change of variables:

$$|q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z}| = |p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}|$$

- This implies

$$\begin{aligned}\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] &= \nabla_{\boldsymbol{\nu}} \int f(\boldsymbol{z}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} = \nabla_{\boldsymbol{\nu}} \int f(\boldsymbol{z}(\boldsymbol{\epsilon})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} \\ &= \nabla_{\boldsymbol{\nu}} \int f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} = \int \nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}\end{aligned}$$

# Reparametrization Trick (2)

$$\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]$$

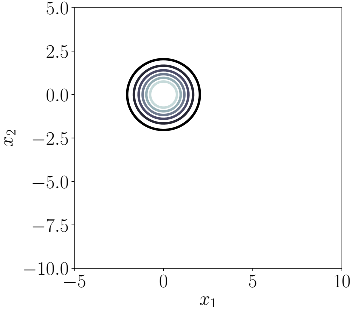
- Change of variables ⏩ Probability mass contained in a differential area must be invariant under change of variables:

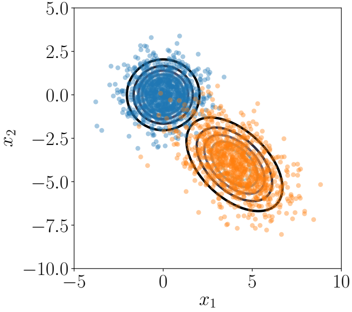
$$|q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z}| = |p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}|$$

- This implies

$$\begin{aligned}\nabla_{\boldsymbol{\nu}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\nu})}[f(\boldsymbol{z})] &= \nabla_{\boldsymbol{\nu}} \int f(\boldsymbol{z}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} = \nabla_{\boldsymbol{\nu}} \int f(\boldsymbol{z}(\boldsymbol{\epsilon})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} \\ &= \nabla_{\boldsymbol{\nu}} \int f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} = \int \nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} \\ &= \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} f(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}))]\end{aligned}$$

# Example



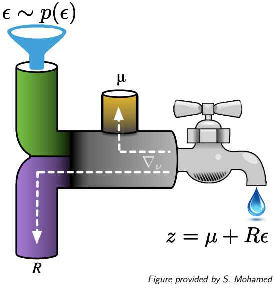

$$\boldsymbol{\nu} := \{\boldsymbol{\mu}, \boldsymbol{R}\}\,, \quad \boldsymbol{R}\boldsymbol{R}^\top = \boldsymbol{\Sigma}$$

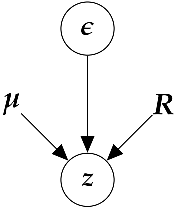
$$p(\boldsymbol{\epsilon}) = \mathcal{N}\big(\boldsymbol{0},\, \boldsymbol{I}\big)$$

$$t(\boldsymbol{\epsilon}, \boldsymbol{\nu}) = \boldsymbol{\mu} + \boldsymbol{R}\boldsymbol{\epsilon}$$

$$\implies p(\boldsymbol{z}) = \mathcal{N}\big(\boldsymbol{z}\,|\,\boldsymbol{\mu},\, \boldsymbol{\Sigma}\big)$$

# Reparametrization as a System of Pipes




- Path: Follow the input noise through the pipes[2]

[2] `https://tinyurl.com/hyakoj2`

# Reparametrization as a System of Pipes

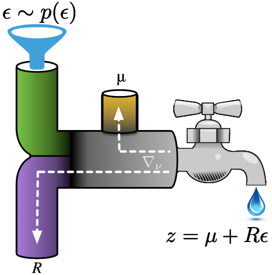
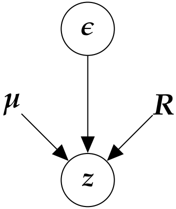



*Figure provided by S. Mohamed*




- Path: Follow the input noise through the pipes[2]
- Construction of pipes known (deterministic transformations)
  - Go back and push gradients through it

[2] `https://tinyurl.com/hyakoj2`

# Reparametrization as a System of Pipes

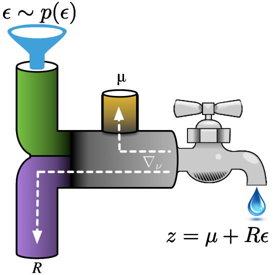
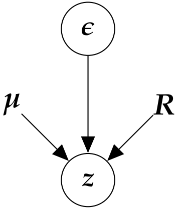


Figure provided by S. Mohamed




- Path: Follow the input noise through the pipes[2]
- Construction of pipes known (deterministic transformations)
  ⏩ Go back and push gradients through it
- Also called "push-in method": Push the parameters of the $z$ distribution into the deterministic transformations

[2] `https://tinyurl.com/hyakoj2`

# Gradients of Expectations: Approach 2

$$\nabla_{\boldsymbol{\nu}} \text{ELBO} = \nabla_{\boldsymbol{\nu}} \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})]$$

# Gradients of Expectations: Approach 2

$$\begin{aligned}\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \nabla_{\boldsymbol{\nu}}\mathbb{E}_q[g(\boldsymbol{z},\boldsymbol{\nu})] \\ &= \nabla_{\boldsymbol{\nu}}\int g(\boldsymbol{z},\boldsymbol{\nu})q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z}\end{aligned}$$

# Gradients of Expectations: Approach 2

$$\begin{aligned}\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \nabla_{\boldsymbol{\nu}}\mathbb{E}_q[g(\boldsymbol{z},\boldsymbol{\nu})]\\ &= \nabla_{\boldsymbol{\nu}}\int g(\boldsymbol{z},\boldsymbol{\nu})q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z}\\ &= \nabla_{\boldsymbol{\nu}}\int g(\boldsymbol{z}|\boldsymbol{\nu})p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}\end{aligned}$$

$$\boxed{q(\boldsymbol{z})d\boldsymbol{z} = p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}}$$

# Gradients of Expectations: Approach 2

$$\begin{aligned}\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \nabla_{\boldsymbol{\nu}}\mathbb{E}_q[g(\boldsymbol{z},\boldsymbol{\nu})] \\ &= \nabla_{\boldsymbol{\nu}}\int g(\boldsymbol{z},\boldsymbol{\nu})q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z} \\ &= \nabla_{\boldsymbol{\nu}}\int g(\boldsymbol{z}|\boldsymbol{\nu})p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon} && \boxed{q(\boldsymbol{z})d\boldsymbol{z} = p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}} \\ &= \nabla_{\boldsymbol{\nu}}\int g(t(\boldsymbol{\epsilon},\boldsymbol{\nu}),\boldsymbol{\nu})p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon} && \boxed{\boldsymbol{z} = t(\boldsymbol{\epsilon},\boldsymbol{\nu})}\end{aligned}$$

# Gradients of Expectations: Approach 2

$$
\begin{aligned}
\nabla_{\boldsymbol{\nu}} \text{ELBO} &= \nabla_{\boldsymbol{\nu}} \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] \\
&= \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z}, \boldsymbol{\nu}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} \\
&= \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z}|\boldsymbol{\nu}) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} && \boxed{q(\boldsymbol{z}) d\boldsymbol{z} = p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}} \\
&= \nabla_{\boldsymbol{\nu}} \int g(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}), \boldsymbol{\nu}) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} && \boxed{\boldsymbol{z} = t(\boldsymbol{\epsilon}, \boldsymbol{\nu})} \\
&= \int \nabla_{\boldsymbol{\nu}} g(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}), \boldsymbol{\nu}) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} && \boxed{\nabla_{\boldsymbol{\nu}} \textstyle\int_{\boldsymbol{\epsilon}} = \int_{\boldsymbol{\epsilon}} \nabla_{\boldsymbol{\nu}}}
\end{aligned}
$$

# Gradients of Expectations: Approach 2

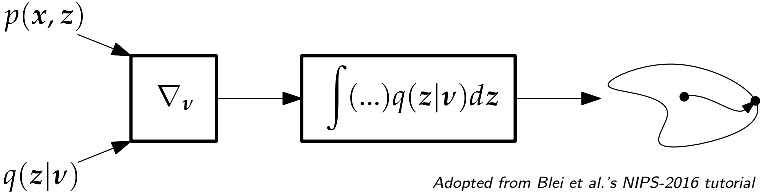
$$
\begin{aligned}
\nabla_{\boldsymbol{\nu}}\text{ELBO} &= \nabla_{\boldsymbol{\nu}}\mathbb{E}_q[g(\boldsymbol{z},\boldsymbol{\nu})] \\
&= \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z},\boldsymbol{\nu})q(\boldsymbol{z}|\boldsymbol{\nu})d\boldsymbol{z} \\
&= \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z}|\boldsymbol{\nu})p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon} && \boxed{q(\boldsymbol{z})d\boldsymbol{z} = p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}} \\
&= \nabla_{\boldsymbol{\nu}} \int g(t(\boldsymbol{\epsilon},\boldsymbol{\nu}),\boldsymbol{\nu})p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon} && \boxed{\boldsymbol{z} = t(\boldsymbol{\epsilon},\boldsymbol{\nu})} \\
&= \int \nabla_{\boldsymbol{\nu}} g(t(\boldsymbol{\epsilon},\boldsymbol{\nu}),\boldsymbol{\nu})p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon} && \boxed{\nabla_{\boldsymbol{\nu}} \textstyle\int_{\boldsymbol{\epsilon}} = \int_{\boldsymbol{\epsilon}} \nabla_{\boldsymbol{\nu}}} \\
&= \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} g(t(\boldsymbol{\epsilon},\boldsymbol{\nu}),\boldsymbol{\nu})]
\end{aligned}
$$

# Gradients of Expectations: Approach 2

$$
\begin{aligned}
\nabla_{\boldsymbol{\nu}} \text{ELBO} &= \nabla_{\boldsymbol{\nu}} \mathbb{E}_q[g(\boldsymbol{z}, \boldsymbol{\nu})] \\
&= \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z}, \boldsymbol{\nu}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z} \\
&= \nabla_{\boldsymbol{\nu}} \int g(\boldsymbol{z}|\boldsymbol{\nu}) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} && \boxed{q(\boldsymbol{z}) d\boldsymbol{z} = p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}} \\
&= \nabla_{\boldsymbol{\nu}} \int g(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}), \boldsymbol{\nu}) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} && \boxed{\boldsymbol{z} = t(\boldsymbol{\epsilon}, \boldsymbol{\nu})} \\
&= \int \nabla_{\boldsymbol{\nu}} g(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}), \boldsymbol{\nu}) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} && \boxed{\nabla_{\boldsymbol{\nu}} \textstyle\int_{\boldsymbol{\epsilon}} = \int_{\boldsymbol{\epsilon}} \nabla_{\boldsymbol{\nu}}} \\
&= \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} g(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}), \boldsymbol{\nu})]
\end{aligned}
$$

▶▶ Turned gradient of an expectation into expectation of a gradient (and sampling from $p(\boldsymbol{\epsilon})$ is very easy).

# Approach




*Adopted from Blei et al.'s NIPS-2016 tutorial*

- Swap order of integration (compute expectations) and differentiation
- **Approximate the expectation after having taken the gradient**

# Pathwise Gradients

$$g(\boldsymbol{z}, \boldsymbol{\nu}) = \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})$$
$$\boldsymbol{z} = t(\boldsymbol{\epsilon}, \boldsymbol{\nu})$$

Simplify gradient of the ELBO:

# Pathwise Gradients

$$g(\boldsymbol{z}, \boldsymbol{\nu}) = \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu})$$
$$\boldsymbol{z} = t(\boldsymbol{\epsilon}, \boldsymbol{\nu})$$

Simplify gradient of the ELBO:

$$\nabla_{\boldsymbol{\nu}} \text{ELBO} = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} g(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}), \boldsymbol{\nu})]$$

$$= \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\nu}} \log p(\boldsymbol{x}, t(\boldsymbol{\epsilon}, \boldsymbol{\nu})) - \nabla_{\boldsymbol{\nu}} \log q(t(\boldsymbol{\epsilon}, \boldsymbol{\nu})|\boldsymbol{\nu})] \quad \boxed{\text{Def. of } g}$$

$$= \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{z}} \log p(\boldsymbol{x}, \boldsymbol{z}) \nabla_{\boldsymbol{\nu}} t(\boldsymbol{\epsilon}, \boldsymbol{\nu})$$
$$- \nabla_{\boldsymbol{z}} \log q(\boldsymbol{z}|\boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} t(\boldsymbol{\epsilon}, \boldsymbol{\nu}) - \underbrace{\nabla_{\boldsymbol{\nu}} \log q(t(\boldsymbol{\epsilon}, \boldsymbol{\nu})|\boldsymbol{\nu})}_{\text{score}}] \quad \boxed{\text{Chain rule}}$$

$$= \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{z}} \big( \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\nu}) \big) \nabla_{\boldsymbol{\nu}} t(\boldsymbol{\epsilon}, \boldsymbol{\nu})] \quad \boxed{\text{Score property}}$$

- Pathwise gradient
- Reparametrization gradient

# Variance Comparison

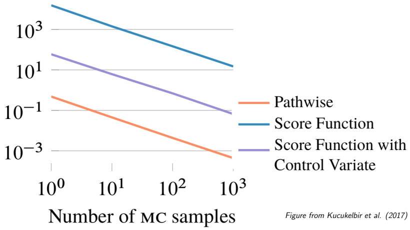



*Figure from Kucukelbir et al. (2017)*

- Drastically reduced variance compared to score-function gradient estimation

# Variance Comparison

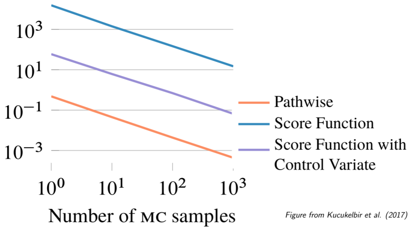



*Figure from Kucukelbir et al. (2017)*

- Drastically reduced variance compared to score-function gradient estimation
- Restricted class of models (compared with score function estimator)

# Score Function vs Pathwise Gradients

$$\text{ELBO} = \int g(\boldsymbol{z}, \boldsymbol{\nu}) q(\boldsymbol{z}|\boldsymbol{\nu}) d\boldsymbol{z}$$

$$g(\boldsymbol{z}, \boldsymbol{\nu}) = \log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\mu})$$

- Score function gradient:

$$\nabla_{\boldsymbol{\nu}} \text{ELBO} = \mathbb{E}_q[\big(\nabla_{\boldsymbol{\nu}} \log q(\boldsymbol{z}|\boldsymbol{\nu})\big) g(\boldsymbol{z}, \boldsymbol{\nu})]$$

  ⏩ Gradient of the variational distribution

- Reparametrization gradient:

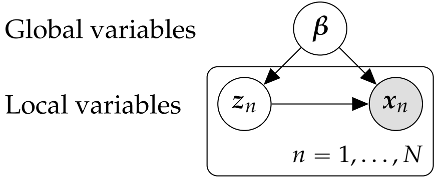
$$\nabla_{\boldsymbol{\nu}} \text{ELBO} = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\big(\nabla_{\boldsymbol{z}} g(\boldsymbol{z}, \boldsymbol{\nu})\big) \nabla_{\boldsymbol{\nu}} t(\boldsymbol{\epsilon}, \boldsymbol{\nu})]$$

  ⏩ Gradient of the model and the variational distribution

# Score Function vs Pathwise Gradients (2)

- Score function
  - Works for all models (continuous and discrete)
  - Works for a large class of variational approximations
  - Variance can be high ⏩ Slow convergence
- Pathwise gradient estimator
  - Requires differentiable models
  - Requires the variational approximation to be expressed as a deterministic transformation $\boldsymbol{z} = t(\boldsymbol{\epsilon}, \boldsymbol{\nu})$
  - Generally lower variance

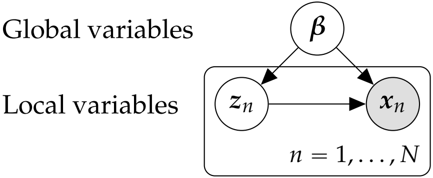
# Re-cap: Hierarchical Bayesian Models




- Joint distribution:

$$p(\boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{x}) = p(\boldsymbol{\beta}) \prod_{n=1}^{N} p(\boldsymbol{z}_n, \boldsymbol{x}_n | \boldsymbol{\beta})$$

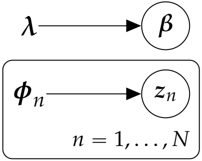
# Re-cap: Hierarchical Bayesian Models




- Joint distribution:

$$p(\boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{x}) = p(\boldsymbol{\beta}) \prod_{n=1}^{N} p(\boldsymbol{z}_n, \boldsymbol{x}_n | \boldsymbol{\beta})$$

- Mean-field variational approximation:



# Re-cap: Mean-Field Stochastic Variational Inference

1. Input: data $\boldsymbol{x}$, model $p(\boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{x})$
2. Initialize global variational parameters $\boldsymbol{\lambda}$ randomly
3. Repeat
   3.1 Sample data point $\boldsymbol{x}_n$ uniformly at random
   3.2 Update local parameter $\boldsymbol{\phi}_n = \mathbb{E}_{\boldsymbol{\lambda}}[...]$
   3.3 Compute intermediate global parameter $\hat{\boldsymbol{\lambda}} = N\mathbb{E}_{\boldsymbol{\phi}_{1:N}}[...] + ...$
   3.4 Set global parameter $\boldsymbol{\lambda} \leftarrow (1-\rho_t)\boldsymbol{\lambda} + \rho_t\hat{\boldsymbol{\lambda}}$

# BBVI Stochastic Variational Inference

1. Input: data $\boldsymbol{x}$, model $p(\boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{x})$
2. Initialize global variational parameters $\boldsymbol{\lambda}$ randomly
3. Repeat
   3.1 Sample data point $\boldsymbol{x}_n$ uniformly at random
   3.2 Update local parameter $\boldsymbol{\phi}_n = \mathbb{E}_{\boldsymbol{\lambda}}[...]$
   3.3 Compute intermediate global parameter $\hat{\boldsymbol{\lambda}} = N\mathbb{E}_{\boldsymbol{\phi}_{1:N}}[...] + ...$
   3.4 Set global parameter $\boldsymbol{\lambda} \leftarrow (1 - \rho_t)\boldsymbol{\lambda} + \rho_t\hat{\boldsymbol{\lambda}}$

Issue:

- Expectations we require to update the local and global parameters are no longer tractable

  ▶▶ No closed-form updating of variational factors

# Addressing the Challenge

- Same problem we had with the ELBO: Integral intractable
  - ⏩ Gradient descent for variational updates

# Addressing the Challenge

- Same problem we had with the ELBO: Integral intractable
  ⏩ Gradient descent for variational updates
- Idea: Stochastic optimization

# Addressing the Challenge

- Same problem we had with the ELBO: Integral intractable
  ⏩ Gradient descent for variational updates
- Idea: Stochastic optimization
- Expectations for updating local variational parameters are computed per data point
  ⏩ Need to run an optimization algorithm per data point
  ⏩ **SVI gets really slow**

# Addressing the Challenge

- Same problem we had with the ELBO: Integral intractable
  ⏩ Gradient descent for variational updates
- Idea: Stochastic optimization
- Expectations for updating local variational parameters are computed per data point
  ⏩ Need to run an optimization algorithm per data point
  ⏩ **SVI gets really slow**

⏩ **Amortized Inference**

# Overview

# Amortized (=shared) Inference

- **Key idea:** Learn a mapping (with global variational parameters) from data points $\boldsymbol{x}_n$ to local variational parameters $\boldsymbol{\phi}_n$:

$$f(\boldsymbol{x}_n, \boldsymbol{\theta}) = \boldsymbol{\phi}_n$$

# Amortized (=shared) Inference

- **Key idea:** Learn a mapping (with global variational parameters) from data points $\boldsymbol{x}_n$ to local variational parameters $\boldsymbol{\phi}_n$:

$$f(\boldsymbol{x}_n, \boldsymbol{\theta}) = \boldsymbol{\phi}_n$$

- No more local variational parameters to optimize

# Amortized (=shared) Inference

- **Key idea:** Learn a mapping (with global variational parameters) from data points $\boldsymbol{x}_n$ to local variational parameters $\boldsymbol{\phi}_n$:

$$f(\boldsymbol{x}_n, \boldsymbol{\theta}) = \boldsymbol{\phi}_n$$

- No more local variational parameters to optimize
- No longer independent optimization of variational parameters

# Amortized (=shared) Inference

- **Key idea:** Learn a mapping (with global variational parameters) from data points $\boldsymbol{x}_n$ to local variational parameters $\boldsymbol{\phi}_n$:

$$f(\boldsymbol{x}_n, \boldsymbol{\theta}) = \boldsymbol{\phi}_n$$

- No more local variational parameters to optimize
- No longer independent optimization of variational parameters
- New global variational parameters $\boldsymbol{\theta}$ can be optimized using stochastic gradient descent

# Amortized (=shared) Inference

- **Key idea:** Learn a mapping (with global variational parameters) from data points $\boldsymbol{x}_n$ to local variational parameters $\boldsymbol{\phi}_n$:

$$f(\boldsymbol{x}_n, \boldsymbol{\theta}) = \boldsymbol{\phi}_n$$

- No more local variational parameters to optimize
- No longer independent optimization of variational parameters
- New global variational parameters $\boldsymbol{\theta}$ can be optimized using stochastic gradient descent
- The mapping is often a deep neural network (inference network)

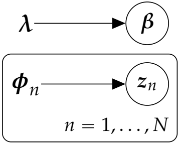
# Amortized (=shared) Inference

- **Key idea:** Learn a mapping (with global variational parameters) from data points $\boldsymbol{x}_n$ to local variational parameters $\boldsymbol{\phi}_n$:

$$f(\boldsymbol{x}_n, \boldsymbol{\theta}) = \boldsymbol{\phi}_n$$

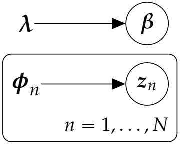
- No more local variational parameters to optimize
- No longer independent optimization of variational parameters
- New global variational parameters $\boldsymbol{\theta}$ can be optimized using stochastic gradient descent
- The mapping is often a deep neural network (inference network)
- Can we overfit? Discuss!

# Amortized VI in Hierarchical Bayesian Models




$$\text{ELBO} = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{\beta}, \boldsymbol{z}_{1:N}) - \log q(\boldsymbol{\beta}, \boldsymbol{z}_{1:N} | \boldsymbol{\lambda}, \boldsymbol{\phi}_{1:N})] \quad \boxed{\boldsymbol{\nu} = \{\boldsymbol{\beta}, \boldsymbol{\phi}_{1:N}\}}$$

# Amortized VI in Hierarchical Bayesian Models

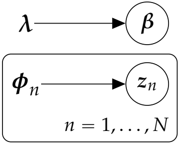



$$\begin{aligned}\text{ELBO} &= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{\beta}, \boldsymbol{z}_{1:N}) - \log q(\boldsymbol{\beta}, \boldsymbol{z}_{1:N} | \boldsymbol{\lambda}, \boldsymbol{\phi}_{1:N})] \\ &= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{\beta}, \boldsymbol{z}_{1:N})] \\ &\quad - \mathbb{E}_q[\log q(\boldsymbol{\beta}|\boldsymbol{\lambda}) + \sum_{n=1}^{N} \log q(\boldsymbol{z}_n | \boldsymbol{\phi}_n)]\end{aligned}$$

$\boldsymbol{\nu} = \{\boldsymbol{\beta}, \boldsymbol{\phi}_{1:N}\}$

mean-field

# Amortized VI in Hierarchical Bayesian Models




$$\begin{aligned}
\text{ELBO} &= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{\beta}, \boldsymbol{z}_{1:N}) - \log q(\boldsymbol{\beta}, \boldsymbol{z}_{1:N} | \boldsymbol{\lambda}, \boldsymbol{\phi}_{1:N})] \\
&= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{\beta}, \boldsymbol{z}_{1:N})] \\
&\quad - \mathbb{E}_q[\log q(\boldsymbol{\beta}|\boldsymbol{\lambda}) + \sum_{n=1}^{N} \log q(\boldsymbol{z}_n | \boldsymbol{\phi}_n)] \\
&= \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{\beta}, \boldsymbol{z}_{1:N})] \\
&\quad - \mathbb{E}_q[\log q(\boldsymbol{\beta}|\boldsymbol{\lambda}) + \sum_{n=1}^{N} \log q(\boldsymbol{z}_n | f(\boldsymbol{x}_n, \boldsymbol{\theta}))]
\end{aligned}$$

$\boldsymbol{\nu} = \{\boldsymbol{\beta}, \boldsymbol{\phi}_{1:N}\}$

mean-field

$\boldsymbol{\phi}_n = f(\boldsymbol{x}_n, \boldsymbol{\theta})$

# Stochastic Gradients

$$\nabla_{\boldsymbol{\theta}}\text{ELBO} = \frac{\text{dELBO}}{\text{d}\boldsymbol{\phi}_n}\frac{\text{d}\boldsymbol{\phi}_n}{\text{d}\boldsymbol{\theta}}$$

- ELBO gradient w.r.t. local variational parameters is difficult
  ▶▶ Stochastic gradient estimators (score function, reparametrization)
- Gradient of variational parameters w.r.t. parameters $\boldsymbol{\theta}$ of inference network are easy

# Amortized SVI

1. Input: data $\boldsymbol{x}$, model $p(\boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{x})$
2. Initialize global variational parameters $\boldsymbol{\lambda}$ randomly
3. Repeat
   3.1 Sample $\boldsymbol{\beta} \sim q(\boldsymbol{\beta}|\boldsymbol{\lambda})$
   3.2 Sample data point $\boldsymbol{x}_n$ uniformly at random
   3.3 Compute stochastic natural gradients

$$\tilde{\nabla}_{\boldsymbol{\lambda}}\text{ELBO}$$

$$\tilde{\nabla}_{\boldsymbol{\theta}}\text{ELBO}$$

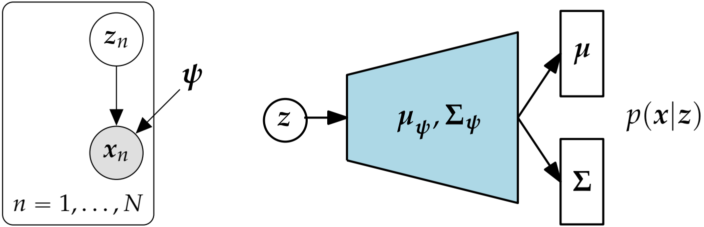
   3.4 Update global parameters

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \rho_t \tilde{\nabla}_{\boldsymbol{\lambda}}\text{ELBO}$$ global variational parameters

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \rho_t \tilde{\nabla}_{\boldsymbol{\theta}}\text{ELBO}$$ inference network parameters

## Example: Variational Auto-Encoder

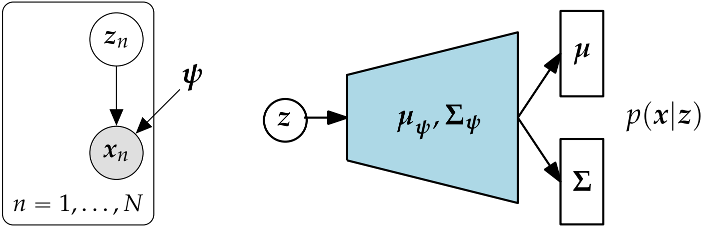
# Variational Auto-Encoder: Model




- Model (Rezende et al., 2014; Kinga & Welling, 2014):

$$p(z) = \mathcal{N}\big(\mathbf{0},\, \mathbf{I}\big)$$
$$p(x|z) = \mathcal{N}\big(\mu_\psi(z),\, \Sigma_\psi(z)\big)$$

# Variational Auto-Encoder: Model

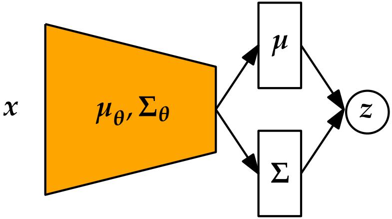



- Model (Rezende et al., 2014; Kinga & Welling, 2014):

$$p(z) = \mathcal{N}\big(\mathbf{0},\, \boldsymbol{I}\big)$$
$$p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{\psi}}(\boldsymbol{z}),\, \boldsymbol{\Sigma}_{\boldsymbol{\psi}}(\boldsymbol{z})\big)$$

- $\boldsymbol{\mu}_{\boldsymbol{\psi}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\psi}}$ are deep networks with model parameters $\boldsymbol{\psi}$

# Variational Auto-Encoder: Inference

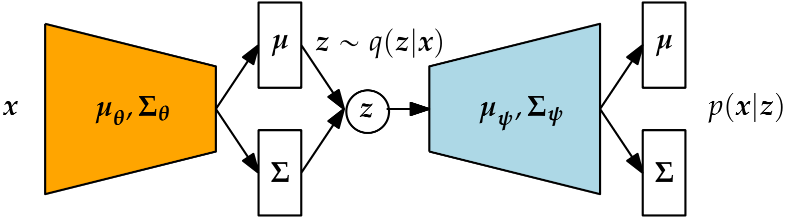



- Inference:

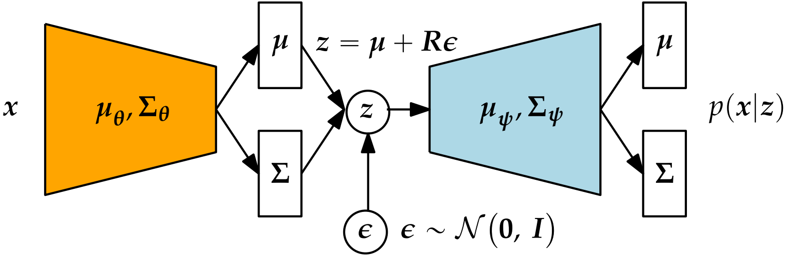
$$q(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{x})\big)$$

- $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ are deep networks that map data onto (local) variational parameters ▸▸ Inference network

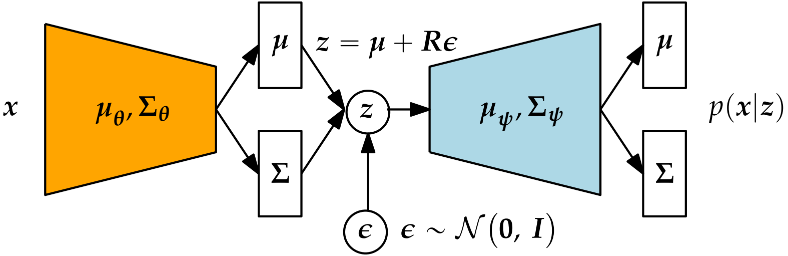
# Variational Auto-Encoder



# Variational Auto-Encoder



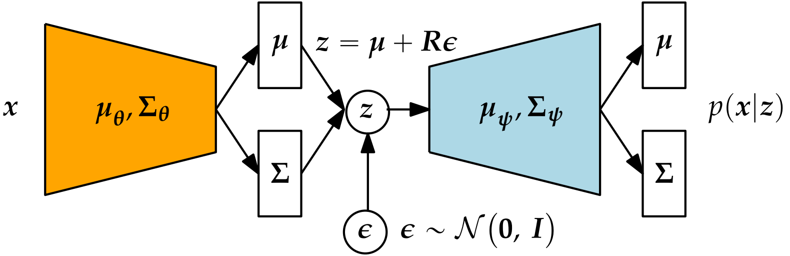
# Variational Auto-Encoder




- Reparametrization trick introduces random noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, but everything else are deterministic transformations

  ⏩ No need to push gradients through samples

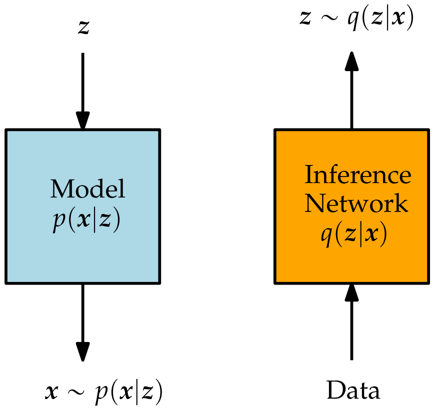
# Variational Auto-Encoder




- Reparametrization trick introduces random noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, but everything else are deterministic transformations

  ⏩ No need to push gradients through samples
- Significance of the VAE:
  - Propagation of gradients through probability distributions (stochastic backpropagation)
  - Joint learning of model parameters and variational parameters

# Variational Auto-Encoder: A Different Schematic




- Generative process (left) (generator)
- Inference process (right) (recognition/inference network)

# Applications

- Data compression/dimensionality reduction (similar to PCA)
- Data visualization
- Generation of new (realistic) data
- Denoising
- Probabilistic data imputation (fill gaps in data)

# VAE: Data Visualization

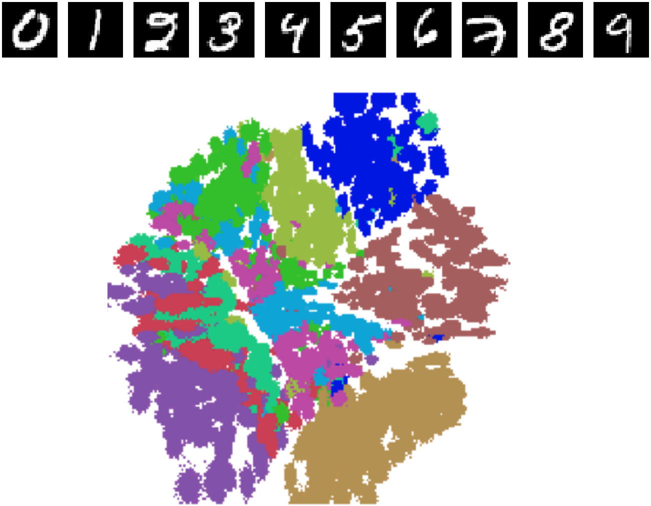


*Figure from Rezende et al. (2014)*

# VAE: Generation of Realistic Images

*Figure from Rezende et al. (2014)*

# VAE: Probabilistic Data Imputation

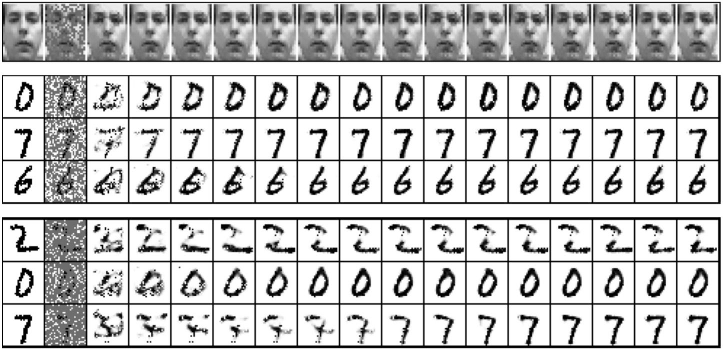


*Figure from Rezende et al. (2014)*

# Overview

| | | Variational approximation of posterior | |
|---|---|---|---|
| | | mean-field | more general |
| Model | conditionally conjugate | analytic solution | |
| | hierarchical Bayesian | stochastic gradient estimators; Example: VAE | dense Gaussian, mixture models, normalizing flows, auxiliary-variable models |

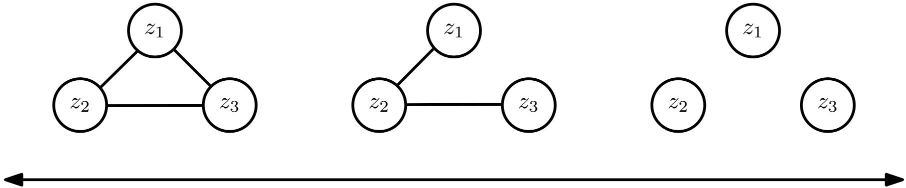
# Overview

# Richer Posterior Approximations


- Build richer posteriors
- Maintain computational efficiency and scalability
- Use all the things we know for specifying models of the data (but now for posterior approximations)

# Structured Mean Field

True posterior | Structured approximation | Fully factorized

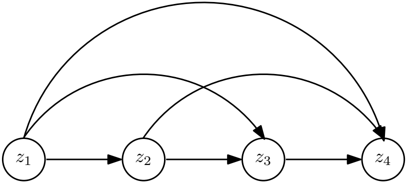


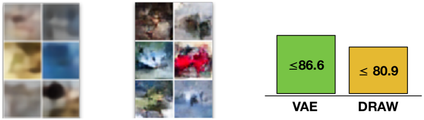

Most expressive | Least expressive

$q(z|x) = p(z|x)$ | $q(z) = \prod_k q_k(z_k|z_{j\neq k})$ | $q(z|x) = \prod_i q_i(z_i)$

- Introduce dependencies between latent variables
  ⏩ Richer posterior than fully factorized

# Autoregressive Distributions




[Gregor et al., 2015]

- Autoregressive structure:

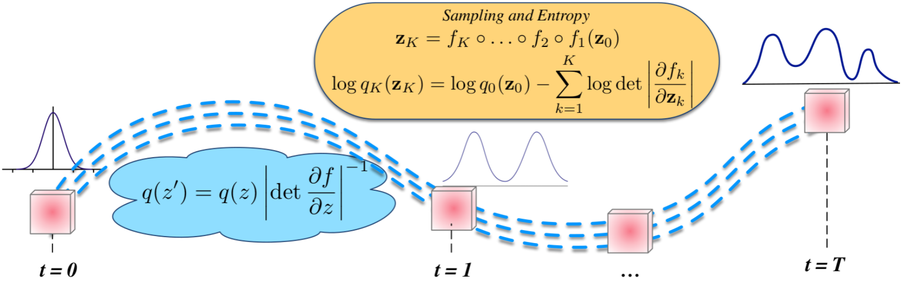
$$q(\boldsymbol{z}_{1:K}|\boldsymbol{\nu}) = \prod_{k=1}^{K} q(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}, \boldsymbol{\nu}_k)$$

- Ordering of latent variables and nonlinear dependencies
- VAE (Rezende et al., 2014): mean-field Gaussian; DRAW (Gregor et al., 2015): autoregressive

# Other Posteriors

- Mixture models (Saul & Jordan, 1996): $q(\boldsymbol{z}) = \sum_k \pi_k q_k(\boldsymbol{z}_k|\boldsymbol{\nu}_k)$
- Linking functions (Ranganath et al., 2016):
  $q(\boldsymbol{z}) = \left(\prod_{k=1}^{K} q(\boldsymbol{z}_k|\boldsymbol{\nu}_k)\right) L(\boldsymbol{z}_{1:K}|\boldsymbol{\nu}_{K+1})$

# Normalizing Flows (1)

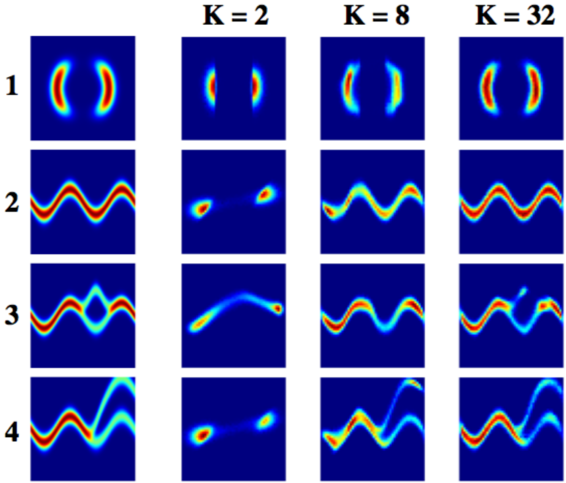



***Distribution flows through a sequence of invertible transforms***

[Rezende and Mohamed, 2015]

- Apply sequence of $K$ invertible transformations to an initial distribution $q_0$ ⏩ Change-of-variables rule

# Normalizing Flows (2)




*Rezende & Mohamed (2015)*

# Summary

- Variational inference finds an approximate posterior by optimization
- Minimizing the KL divergence is equivalent to maximizing a lower bound on the marginal likelihood
- Mean-field VI: analytic updates in conditionally conjugate models
- Stochastic VI: Stochastic optimization for scalability
- General models require us to compute gradients of expectations
  - Score-function gradients
  - Pathwise gradients
- Amortized inference
- Modern VI allows us to specify rich classes of posterior approximations

# References I


[1] M. J. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.

[2] C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer-Verlag, 2006.

[3] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[4] S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, and G. E. Hinton. Attend, Infer, Repeat:Fast Scene Understanding with Generative Models. In *Advances in Neural Information Processing Systems*, 2016.

[5] Z. Ghahramani and M. J. Beal. Propagation Algorithms for Variational Bayesian Learning. In *Advances in Neural Information Processing Systems*, 2001.

[6] P. Gopalan, W. Hao, D. M. Blei, and J. D. Storey. Scaling Probabilistic Models of Genetic Variation to Millions of Humans. *Nature Genetics*, 48(12):1587–1590, 2016.

[7] P. K. Gopalan and D. M. Blei. Efficient Discovery of Overlapping Communities in Massive Networks. *Proceedings of the National Academy of Sciences*, page 201221839, 2013.

[8] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra. DRAW: A Recurrent Neural Network For Image Generation. In *Proceedings of the International Conference on Machine Learning*, 2015.

[9] M. D. Hoffman, D. M. Blei, and F. Bach. Online Learning for Latent Dirichlet Allocation. *Advances in Neural Information Processing Systems*, 23:1–9, 2010.

[10] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic Variational Inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

[11] D. Jimenez Rezende and S. Mohamed. Variational Inference with Normalizing Flows. In *Proceedings of the International Conference on Machine Learning*, 2015.

[12] D. Jimenez Rezende, S. Mohamed, and D. Wierstra. Stochastic Backpropagation and Variational Inference in Deep Latent Gaussian Models. In *Proceedings of the International Conference on Machine Learning*, 2014.

# References II

[13] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37:183–233, 1999.

[14] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *Proceedings of the International Conference on Learning Representations*, 2014.

[15] A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, and D. M. Blei. Automatic Differentiation Variational Inference. *Journal of Machine Learning Research*, 18(1):430–474, 2017.

[16] J. R. Manning, R. Ranganath, K. A. Norman, and D. M. Blei. Topographic factor analysis: A bayesian model for inferring brain networks from neural data. *PloS One*, 9(5):e94914, 2014.

[17] T. P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2001.

[18] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge, MA, USA, 2012.

[19] R. Ranganath, D. Tran, and D. M. Blei. Hierarchical Variational Models. In *Proceedings of the International Conference on Machine Learning*, 2016.

[20] H. Salimbeni and M. P. Deisenroth. Doubly Stochastic Variational Inference for Deep Gaussian Processes. In *Advances in Neural Information Processing Systems*, 2017.

[21] L. K. Saul and M. I. Jordan. Exploiting Tractable Substructures in Intractable Networks. In *Advances in Neural Information Processing Systems*, 1996.

[22] R. J. Williams. Simple Statistical Gradient-following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8(3):229–256, May 1992.